Data Mining and Knowledge Discovery (2023) 37:1–38
https://doi.org/10.1007/s10618-022-00871-y

Practical joint human-machine exploration of industrial time series using the matrix profile

Felix Nilsson[1] · **Mohamed-Rafik Bouguelia**[2] · **Thorsteinn Rögnvaldsson**[2]

Received: 6 December 2021 / Accepted: 11 September 2022 / Published online: 5 October 2022

Abstract

Technological advancements and widespread adaptation of new technology in industry have made industrial time series data more available than ever before. With this development grows the need for versatile methods for mining industrial time series data. This paper introduces a practical approach for joint human-machine exploration of industrial time series data using the Matrix Profile, and presents some challenges involved. The approach is demonstrated on three real-life industrial data sets to show how it enables the user to quickly extract semantic information, detect cycles, find deviating patterns, and gain a deeper understanding of the time series. A benchmark test is also presented on ECG (electrocardiogram) data, showing that the approach works well in comparison to previously suggested methods for extracting relevant time series motifs.

Keywords Time series · Matrix profile · Motif discovery · Industry 4.0

Responsible editor: Eamonn Keogh

Mohamed-Rafik Bouguelia and Thorsteinn Rögnvaldsson these authors contributed equally to this work.

✉ Felix Nilsson
fenil@hms.se

Mohamed-Rafik Bouguelia
mohamed-rafik.bouguelia@hh.se

Thorsteinn Rögnvaldsson
thorsteinn.rognvaldsson@hh.se

[1] HMS Labs, HMS Industrial Networks AB, Stationsgatan 37, 30250 Halmstad, Sweden

[2] Center for Applied Intelligent Systems Research, Halmstad University, Kristian IV:s väg 3, 30118 Halmstad, Sweden

1 Introduction

The introduction of Industry 4.0 will connect everything on the industry floor to the cloud and the Industrial Internet of Things. Significant amounts of data are already collected today and to some extent used in Supervisory Control And Data Acquisition (SCADA) software for high-level monitoring and control of industrial and manufacturing processes. Everything on the factory floor of tomorrow will be connected and communicating with IT systems worldwide, which will generate immense amounts of time series data.

With this development grows the need for versatile methods for mining industrial time series data, to find interesting subsequences in the data. The word "interesting" in this context might mean patterns that are repeated once or several times. It could also be patterns that occur seldomly. The terms *time series motif* and *time series discord* have been used to define such patterns in the literature (Lin et al. 2002; Yeh et al. 2016). Motifs are defined as the subsequences with the lowest distance to their nearest neighbors within a time series. Discords are, in contrast, the subsequences that have the highest dissimilarity to all other subsequences. Motifs and discords in industrial time series data can provide valuable hints on the state of a product or process, such as: What is it doing? Is it operating as it should? Is it operating efficiently?

The Matrix Profile (MP) method (Yeh et al. 2016), finds motifs much faster and with fewer hyper-parameters than any previous approach. With it, it should be possible to approach real-world large-scale industrial problems to find motifs that can help operators and machine builders to better explore and understand their processes, machines, and products. However, there are several practical issues to deal with, e.g., limited computing resources, variable-length motifs, finding motifs invariant of scale, warping or other kinds of issues. Some of these challenges might be solved conveniently in an environment where a human operator (the expert) works together with a motif discovery tool and iteratively explores the time series. This paper builds upon this idea and proposes an approach for human-machine collaboration on time series data exploration and analysis using the MP. The intention is to let users extract meaningful knowledge from their time series data by discovering interesting motifs and clustering them into (relatively) few representatives. This is done without necessarily knowing beforehand what the data is expected to contain (e.g., in terms of class labels), which is usually the case in real production scenarios. Several important practical issues are encountered and discussed in this work:

- How to handle motifs of variable length or to set an appropriate size for the sliding window required to calculate the MP.
- Industrial processes span over long periods of time and generate large data sets. Most documented results with MP use time series with a length of up to 2 million samples (Yeh et al. 2016; Zhu et al. 2016, 2018), which is at least five times shorter than the industrial data sets used in this paper.
- The expert should have an active role and be able to incorporate domain knowledge or provide guidance to find meaningful patterns and exclude uninteresting subsequences.

- The MP indicates where potential motifs are in the time series, but it does not tell how many there are and whether there are groups of similar motifs. How should relevant motifs be extracted when their numbers are unknown? How should similar motifs be grouped? Can any information be gained from these groups?

The Matrix Profile is a relatively new method, and there are few studies on real-life industrial data that discuss the practical problems encountered with such data. This paper is centered around three real-world case studies to investigate whether an MP-based solution can handle the issues listed above; what other issues there are; what the practical limitations are of the MP; and how an MP-based solution compares to previous research where these data sets have been involved. The contributions of this work are the following:

- The construction and evaluation of a first-generation tool for exploring time series data in collaboration between a human expert and the MP algorithm.
- An overview of practical issues for such a tool to handle to be useful in a real-world application.
- A demonstration of the proposed method on how to extract semantic information from an industrial time series.
- A demonstration of the proposed method on how to detect cycles in an industrial time series.

2 Related work

Motif discovery has been applied in many domains, from entomology, meteorology, geology, medicine, engineering, biology, music, to finance (Mueen 2014; Torkamani and Lohweg 2017). Motif discovery algorithms can be grouped into two categories: exact and approximate. It is often desirable to calculate the exact motifs, but that usually comes at the cost of increased computing complexity. Typical methods of reducing complexity are Random Projection (Chiu et al. 2003), Piecewise Aggregate Approximation (PAA), Symbolic Aggregate approXimation (SAX), and Discrete Fourier Transform (DFT) (Mueen 2014). These representations reduce the dimensionality on which to calculate the motifs and thereby reduce the complexity. However, the trade-off is typically a loss of information. Early algorithms like Enumeration of Motifs through Matrix Approximation (EMMA) (Lin et al. 2002) use PAA to reduce the continuous time series into a discrete set of equiprobable symbols. Later approaches often build on SAX. One example is the MK algorithm by (Chiu et al. 2003), which converts all subsequences using SAX and then randomly projects them into a lower-dimensional space and measures the collisions between subsequences. The subsequence with the most collisions becomes the best motif, the one with the second most collisions becomes the second best, and so on. SAX has been the most cited time series representation for a long time (Torkamani and Lohweg 2017). The most common distance measure in motif discovery algorithms is the Euclidean distance, although many other distance measures exist like Dynamic Time Warping (DTW) and Uniform scaling. The Euclidean distance's popularity stems from it being the most intuitive and the fastest to compute. The Euclidean distance's main drawback is that it

deals poorly with differences in amplitude, scaling, and different length subsequences. Nevertheless, it has proven to be competitive against more advanced distance measures (Renard 2017). Also, Mueen (2014) argues that the resulting distances between motifs are practically indistinguishable between using DTW or Euclidean distance.

In addition to the methods above, various data mining and machine learning techniques have been used to discover motifs. Everything, from clustering, to Minimum Length Descriptors and PCA to reduce the dimensionality of multivariate data into a single time series (Tanaka et al. 2005), to image representations of motifs and Fully Convolutional Networks. The latest significant contribution to time series motif discovery is the Matrix Profile, which was first published in 2016, and is arguably the current state-of-the-art method for motif discovery in time series.

Motif discovery methods return information about where motifs are located in a time series and the location of a motif's nearest neighbor. Information about groups of similar motifs or how many motifs there are in each group can be provided by motif enumeration methods like Set Finder, Scan MK (Bagnall et al. 2014), and HubFinder (Yoshimura et al. 2019). The main practical problem with these algorithms is that they require hyper-parameters that are not easy to set when working with large industrial data sets. Set Finder and Scan MK require the user to select a threshold radius that dictates how motifs are merged into clusters. This threshold has been documented to be sensitive and difficult to set (Yoshimura et al. 2019). HubFinder does not require a radius but the user must decide how many motif categories to extract by setting a hyper-parameter K (Yoshimura et al. 2019). Unfortunately, on industrial data sets that potentially contain several years of process data, it is not easy to know in advance how many motif categories to extract.

Keogh and Lin (2005) show that clustering time series subsequences is meaningless. However, the paper explains that this meaninglessness results from the way the overlapping subsequences are extracted via a sliding window. Instead of attempting to cluster every subsequence, including trivial matches, the authors suggest a potential solution to cluster subsequences meaningfully: first, running a motif detection algorithm to extract an initial set of promising subsequences, and then clustering only these subsequences. The approach we propose in this paper is in line with this solution.

The proposed method described in Sect. 4 does not require the user to define a radius or decide how many patterns to extract beforehand. Instead, it relies on a joint human-machine exploration process where new knowledge is created systematically.

3 Data

Three industrial data sets are used in the experiments and comparisons, all collected from real-world processes. We also use a medical time series from the MIT-BIH Arrhythmia Database[1], for the purpose of comparing with other motif enumeration algorithms on labeled data.

[1] https://archive.physionet.org/physiobank/database/html/mitdbdir/mitdbdir.htm

3.1 The oil purification data set

The Oil Purification (OP) data set contains data from an oil separation process where oil is separated from dirt and unwanted particles. This data set is not labeled and there are no known faults or deviations. The process is precisely repeated with two different types of patterns on two different time scales. There are no missing data in this data set. The data consists of 3 signals from a centrifugal separator machine, sampled at 1 Hz, spanning a period of 3,380 hours ($\sim$ 141 days).

3.2 The water treatment data set

The Water Treatment (WT) data set contains data collected from freshwater treatment equipment. The data set is semantically labeled since data is collected from the transitions between states in the water purification process. The goal is to see if the semantics of the state transitions can be seen in the motifs extracted from the time series. The different state transitions are labeled in the data set. The data is not continuous; the time series are segments from the state transition periods. The data consists of 1,244 hours ($\sim$ 51 days) recorded over 13 month.

3.3 The city bus fleet data set

The City Bus Fleet (CBF) data set contains 8,750 hours ($\sim$ 365 days) of air pressure data for a compressed air circuit sampled at an average sampling rate[2] of 3.6 Hz from 19 different city buses in actual operation. The data has many periods with missing values, but it also contains annotations of interesting events like component failures and workshop visits. The air compressor is a key component that is expected to break very seldomly, but if it breaks the bus is impossible to operate. The signal amplitude is expected to be stationary but the frequency varies depending on how much the pressurized air is used. This data set has been used by Fan et al. (2015, 2016, 2020) to predict compressor failures and maintenance needs.

3.4 The ECG arrhythmia data set

The MIT-BIH Arrhythmia Database is available on PhysioNet and contains excerpts from 47 subjects of two-channel ambulatory Electrocardiogram (ECG) recordings (Moody and Mark 2001). We use the signal for subject 106 to compare with previous motif finding studies (Yoshimura et al. 2019). This subject's data has 1507 normal beats and 520 premature ventricular contractions. The upper signal (the modified limb lead II) is used as the univariate time series.

[2] The sampling is uneven and varies between buses.

4 Methodology

This section describes the steps in the proposed human-machine iterative workflow for exploring large industrial time series. First, a short introduction to the MP is provided, followed by some insights from working with it. Then, the first step of the proposed approach, called GAME (Gaussian Assisted Motif Extraction), is presented. This is followed by the presentation of the second step, called IUSE (Iterative User-Assisted Refinement), for improving the motif groups produced by GAME. Finally, the metadata structure produced by IUSE is described, and a summary of the proposed approach is provided.

The proposed process works on top of a pre-calculated MP. The user can therefore use any suitable MP algorithm. Motifs are extracted and, with the aid of the user, grouped into semantically meaningful groups. The first step is to do a rough estimation of the distribution of the MP values. This enables the extraction of groups of patterns from the time series and does not require the user to set a threshold such as "get the top 10 best motifs". The second step is to group the extracted motifs, with assistance from the user, into semantically meaningful clusters. This creates an annotation data structure that can be used either as is to segment the time series, or in further data mining efforts.

4.1 The Matrix Profile

The Matrix Profile is a metadata structure of nearest neighbor distances between subsequences in a time series T. It was introduced in (Yeh et al. 2016) and over 20 additional MP-related publications have been released since. The MP is fast, easy to compute, and memory efficient; it can be calculated in as little as $\mathcal{O}(n^2)$ time (Yeh et al. 2016; Zhu et al. 2018). The simplest way of calculating the MP is by sliding a window of length m across the n-length time series and calculate the distance between this subsequence and all the other m-length subsequences. This produces a two-dimensional matrix of distances. The MP is then produced by saving the smallest distance in each column, excluding the trivial matches (i.e., the self-matches). The indices of the minimum values are also saved in a Matrix Profile Index (MPI). Together, the MP and the MPI store the distance to and the location of the closest matching subsequence to every subsequence. The MP's simplicity allows the calculations to be easily distributed across hardware to further accelerate the computation.

The memory efficiency of the MP algorithms depends on the implementation. The entire $\mathcal{O}(n^2)$ distance matrix cannot be stored in memory since it would require hundreds of gigabytes of memory even for medium sized time series. Different implementations of the MP algorithms manage the memory differently, which is an important aspect to consider when selecting the MP algorithm based on the application. However, the resulting MP only occupies $\mathcal{O}(n)$ memory which is quite manageable even for very large amounts of data. All MP publications and related material can be found on the MP home page[3].

[3] https://www.cs.ucr.edu/~eamonn/MatrixProfile.html

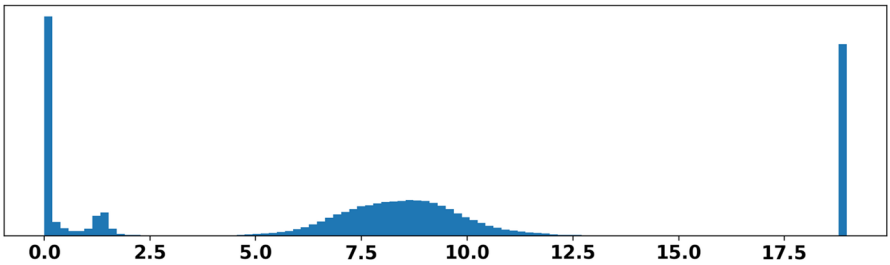

Fig. 1 A histogram of values from a Matrix Profile calculated using sensor data from the Oil Purification data set

The MP distances are z-normalized, which means that sections of data with constant values will cause division by zero type of errors if they are longer than the MP window length. Depending on the implementation, the result might be a NULL or NaN value inserted in those index positions in the MP. Depending on the programming language, this might be interpreted as a very low, or a very high number, or both. Almost constant values, i.e. sections with very small variances, tend to push the distance value high. In situations where the user is looking for discords, simply looking for the indices with the MP's largest values will return near-constant sections. This is particularly a problem in the case of sensor data, where it is common that the sensor monitoring a constant phenomenon produces a jittering signal jumping between one or two points from the mean value. Near constant sections of data can also result in the opposite problem, that the distance is pushed low, since flatter sections can have very good matches to other low variance subsequences. In our experience, single spikes in the distribution of the MP often occur around zero and the MP max value, like in Fig. 1, which is a histogram of the MP values calculated on data from the OP data set described in Sect. 3. The spikes close to zero and above 17 result mostly from either null-sequences or low variance sequences. If a user uses the global minimum of the MP to get a motif, or the global maximum to get a discord, the results will mostly be uninteresting. This is not an unknown problem, for example the latest version of the MERLIN tool for anomaly detection takes measures to avoid these (Nakamura et al. 2020).

The values in the MP say nothing about the frequency of a motif; motifs and discords are only about similarity and dissimilarity to the closest matching subsequence, not about the frequency of occurrence. A motif can have a very good match even if it only happens "once in a blue moon", i.e. very seldomly. A rare motif is thereby not equal to a discord. Similarly, the MP value says nothing about the shape of the motif, only the distance to the most similar subsequence. This means that two subsequence pairs with similar pair-wise distances do not have to be similar motifs. A similarity or dissimilarity between pairs can only be discovered by comparing the motif pairs to each other.

The length m of the sliding window must be set before the MP is computed. The MP is capable of capturing motifs that are a bit shorter than the sliding window (Yeh et al. 2016), thus the window length should be at least the length of a potential pattern. This requires some domain knowledge about the data, but it is usually enough to do

a rough estimation since the MP is forgiving when it comes to window size. It can be beneficial to try a few different window sizes to find one that best corresponds to expected patterns in a time series.

A more systematic approach to estimating m can be to calculate the Pan-Matrix Profile suggested by (Madrid et al. 2019). It is a two-dimensional structure containing nearest neighbor information for window sizes between a lower and an upper threshold, which essentially involves sliding several windows of different sizes across the data. This has also been the approach of other methods like the k-Best Motif Discovery (kBMD) (Nunthanid et al. 2012). The (obvious) downside with this is a high computational cost. The kBMD algorithm has, according to (Nunthanid et al. 2012), a time complexity that is "*quite high*". The Pan-Matrix Profile algorithm, on the other hand, has a time complexity of $\mathcal{O}(n^2 r)$, where r is the number of window sizes, which is similar to calculating one MP per window size with state of the art MP algorithms (Zhu et al. 2018).

4.2 Getting the top motifs

Getting the top k motifs is probably the first approach one thinks of when working with the MP. Algorithm 1, returns the indices for the k lowest values in the MP, excluding m points before and after the motif. The algorithm can easily be changed to find discords by changing lines three and four to argmax and negative infinity. The problem with this "top k" approach is how to set k so that all relevant motifs are found. The issues mentioned earlier mean that this is very difficult and often the outcome is a large number of uninteresting motifs (or discords).

Algorithm 1 Get top k motifs

```
    MP: matrix profile values
    k: number of motifs to extract
    m: window length
1:  procedure GETTOPKMOTIFS(MP, k, m)
2:      for i ← 1...k do
3:          idx ← argmin(MP)
4:          MP[idx − m : idx + 2m] ← ∞
5:          kMotifs[i] ← idx
6:      end for
7:      return kMotifs
8:  end procedure
```

An alternative to the "top k" approach is to do a rough estimation of the MP data distribution using a Gaussian distribution. Then, based on the mean, and standard deviation for that Gaussian, one can calculate the standard score, Equation 1, for all points in the MP. Essentially, this is what GAME (described in Sect. 4.3) does.

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

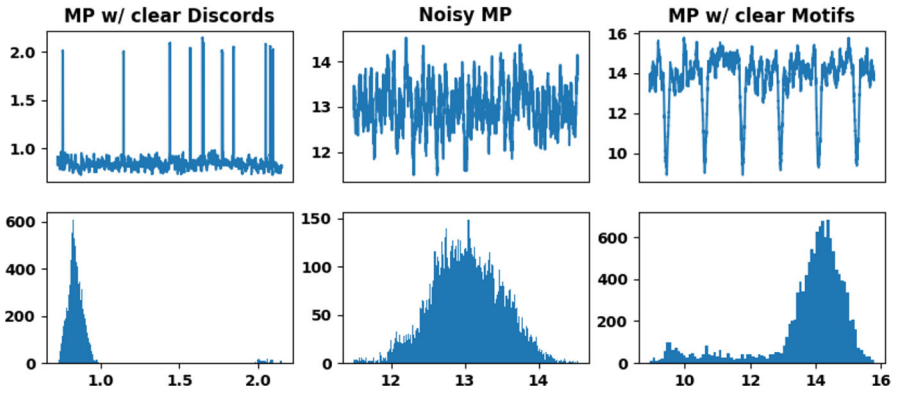

Fig. 2 Three Matrix Profiles (top row) and their corresponding distributions of values (bottom row). The left column is an example of an MP with clear discords, the right column is an example with clear motifs, and the middle column is an MP of uniform noise (i.e. no discords or motifs)

Motifs can then be extracted from the MP by selecting all values with a value one, two, or three standard deviations away from the mean. Retrieving all points with a negative z-score of -2 or less would return roughly the top 2% of the best matching subsequences in the time series. Going two standard deviations in the opposite direction would similarly yield the top 2% of the worst matching subsequences. Admittedly, it cannot be expected that the distribution of MP values will be Gaussian, but, as will be shown later, the core concept can be used on an arbitrary distribution.

4.3 Gaussian Assisted Motif Extraction (GAME)

The first step in GAME is to do a rough estimation of the density distribution by constructing a histogram of the MP values. To illustrate this, Fig. 2 shows three types of MPs from synthetic data, together with histograms of their values. Time series with random noise data will have bell shape distributed MP values. Time series with distinct discords tend to have right-skewed distributions. Time series with distinct motifs will have left-skewed distributions.
MP values from real data will be distributed according to varying mixtures of these three basic types. The z-normalized distance in the MP depends on the number of samples m in the time window so the position of the bell shape will depend on this.

The number of bins used to construct a histogram of the MP is set automatically using the Freedman-Diaconis rule (Freedman and Diaconis 1981), as shown in Equation 2:

$$\begin{aligned} \text{number of bins} &= \left\lceil \frac{\max(\text{MP}) - \min(\text{MP})}{h} \right\rceil \\ h &= 2\frac{\text{IQR}(\text{MP})}{\sqrt[3]{n}} \end{aligned} \tag{2}$$

where n is the Matrix Profile length, and the IQR is the interquartile range defined as the difference between the 75^{th} and 25^{th} percentiles of the data (i.e., the respective values below which 75% and 25% of the distribution lies).

The top row of Fig. 3 shows the distributions of MP values for two industrial time series. They are not Gaussian distributions, but the distributions can be approximated well using a Gaussian Mixture Model (GMM). The GMM fit is done by first finding the peaks in the distribution and then applying the expectation-maximization (EM) algorithm (Dempster et al. 1977). The peaks are found by passing the bin heights through a Savitzky-Golay filter (Savitzky and Golay 1964) using a third-order polynomial to generate a smoother curve, and then using a "watershed" type of algorithm based on persistent homology (Huber 2021). The Python implementation of the peak detection algorithm was taken from Stefan Huber's web-page [4]. The results of the peak detection step are shown in the middle row of Fig. 3, where the grey shows the density distribution (histogram) outlined with a blue edge, the red lines are the smoothed curves, and the dotted black vertical lines are the found peaks. The EM algorithm is then applied with the found number of peaks $\pm k$ components, and the best fit is determined using Akaike's information criterion (Akaike 1974), $AIC = 2k - 2\ln(\hat{L})$. Where k is the number of components and $\hat{L}$ is the maximum value of the GMM's likelihood function. The GMM with the lowest AIC score is used in the next step. The bottom row in Fig. 3 shows examples of the Gaussians found for the distributions in the top row of Fig. 3.

The GMM is used to estimate how likely it is that each point in the MP is drawn from the distribution of a certain Gaussian component, and each point is assigned to its most likely Gaussian. Extraction is done by retrieving all, or part, of the points belonging to a specific Gaussian component. Gaussian components further to the right in the distribution will contain subsequences with a longer distance to their closest match, i.e. discords, and Gaussian components further to the left will contain subsequences with a shorter distance to their closest match, i.e. motifs. Subsequences assigned to the Gaussians in the center are usually neither motifs nor discords. Quickly traversing the Gaussians and visualizing the motifs will give the user an initial intuition about where the motifs are and what they look like. This is then improved throughout the processes that follow (GAME and IUSE).

Nearby points in the MP likely have similar values and would end up in the same Gaussian cluster. The points are grouped into blocks of consecutive indices to avoid retrieving duplicates of the same subsequences but with slight offsets. Motifs are extracted from each block similarly to Algorithm 1 until each block is empty. Measures are taken so that motifs extracted from different blocks do not overlap in time.

Up to this point, a data set of potentially millions of points has been reduced to a few thousand interesting subsequences, which is much more manageable. However, reviewing thousands of motifs can still be overwhelming, even though it is a great reduction in size from the original tens of millions of points. The next step is therefore to group the motifs so they can be presented to the user. The number of clusters (or groups) to consider at this stage is subjective since it depends on the interpretation, need, and interest of the user. Therefore, we don't decide on the exact number of clusters

[4] https://www.sthu.org/blog/13-perstopology-peakdetection/index.html

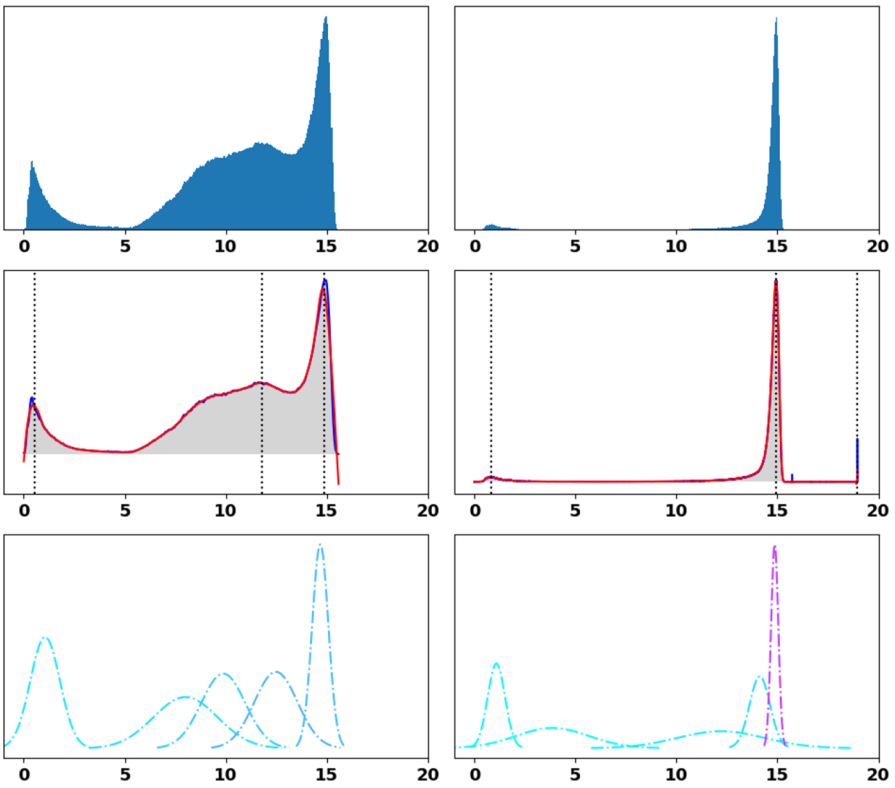

Fig. 3 Histograms of the Matrix Profile calculated for two signals from an industrial data source (top row). Three peaks were detected in each of the signals and are shown in the middle row (marked with black lines). Both distributions could be best approximated using five Gaussian components (bottom row). Darker magenta color indicates a higher weight in the Gaussian mixture

beforehand. Instead, we group the motifs into an initial hierarchy using agglomerative hierarchical clustering, which is a quite fast operation on such few points, and the user can tune the clustering with a simple slider. Later, the user can adjust the clustering result with the help of IUSE (i.e. by merging clusters etc), as described in subsection 4.4. The idea is to generate a good starting point for the IUSE operations that follow. An example result from such initial clustering (before using IUSE) is shown in Fig. 4.

Next, the motifs in each cluster are aligned (within the same cluster) as shown in Algorithm 2; T is the time series, *indices* are the starting indices of the motifs, and m is the motif length. The first motif is selected as the anchor motif on line 2, and all subsequent motifs are aligned with this one. This is to avoid having the motifs drifting into the flat sections of the data set, which might be a risk otherwise. Then, on lines 4-7, distance profiles are calculated between each motif $\pm m$ points and the anchor motif. The adjusted index is appended to the $newIndices$ list on line 7 and returned on line 9.

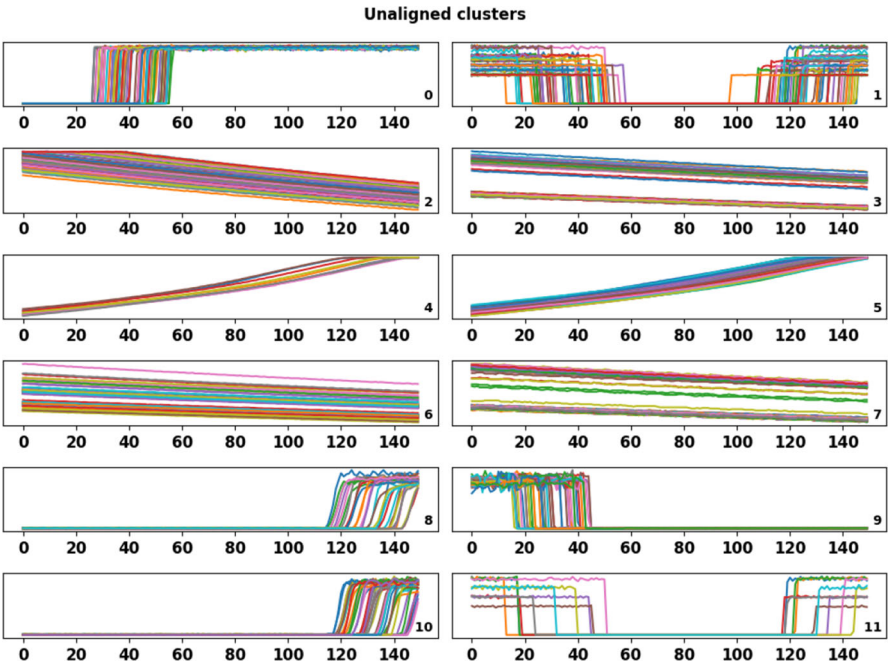

Fig. 4 An example result of the hierarchical motif clustering

Algorithm 2 Align motifs

T: time series
indices: indices of the motifs in the time series
m: window length

```
function ALIGNMOTIF(T, indices, m)
    anchor ← T[indices[0] : indices[0] + m]
    newIndices[0] = indices[0]
    for idx in indices[1 : ] do
        S = T[idx-m:idx + 2m]
        D = MASS(S, anchor)
        newIndices.append(idx - m + argmin(D))
    end for
    return newIndices
end function
```

The MP values grow with the length of the window and the distribution of the MP values therefore also changes. If the window length is very short, e.g. a few samples, then the MP values will be very small (almost every short sample will have a closely matching segment) and the distribution will be similar to that shown in the left panel, bottom row, in Fig. 2. When the window length is very long, then almost all MP values will be large, and the distribution will be similar to that shown in the

right panel, bottom row, in Fig. 2. When the window length is one to a few times the minimum motif length, then there will be visible peaks in the distribution and the number of peaks will be stable. However, as the window length is increased, these peaks will move rightwards. As long as the window length is not much shorter than the minimum motif length, the result of the GAME approach is quite insensitive to the exact value of the window length, especially since the focus is often on peaks to the left in the distribution. Moreover, as presented by Madrid et al. (2019), the MP Euclidean distances are typically stable around a particular subsequence length.

4.4 Iterative User-Assisted Refinement (IUSE)

In the iterative refinement step, the user takes a more active part. Figure 4 displays an example result after agglomerative hierarchical clustering: 12 reasonably homogeneous clusters. After being aligned, the subsequences in the clusters end up as shown in Fig. 5. Some of these contain what seems to be the same pattern. These can be adjusted and merged manually by user selection, after which only six almost homogeneous clusters remain, see Fig. 6.

Along with the cluster assignments, patterns within a group can be differentiated using the concept of "sub-clusters". Sub-clusters might be used to organize patterns in a group and keep a record of different variations of very similar patterns. This can be useful to track how patterns change over time or used to separate deviating motifs from the main groups. The sub-cluster labels can be extracted by applying the previous steps again to a specific cluster. All the operations described under this section are relatively computationally inexpensive to perform. It is therefore possible for a user to work with IUSE in an iterative exploratory manner.

There are several places in the process where the user has the opportunity, or is required, to contribute to the extraction and clustering process. These are described in more detail below:

First, the user needs to set a suitable window size m for the MP calculation. The GAME process works on top of an MP, and the MP calculation is the most expensive step in terms of processing time. The user also needs to select which MP implementation that is best suitable for the particular problem. If "anytime" behavior is desired, an algorithm like SCRIMP++ (Zhu et al. 2018) makes sense. This is currently the fastest anytime algorithm to calculate the MP. If the motifs vary in length, a suitable MP algorithm is SWAMP (Alaee et al. 2020), which calculates the MP using Dynamic Time Warping instead of Euclidean distance.

The EM algorithm produces a GMM and the user decides which Gaussian to explore. Depending on the density distribution and the goal for the search, this may vary. In all examples this far, the goal has been to find motifs and the left-most Gaussian is used. However, the density estimation and partitioning into Gaussians divide the patterns into natural groups and, from a knowledge gathering perspective, the user can benefit from traversing the groups one by one.

The user has a number of tools available once patterns are extracted via their Gaussian assignments. The first one is to select where to prune the hierarchical tree produced by the agglomerative hierarchical clustering. The clustering and pruning operations are

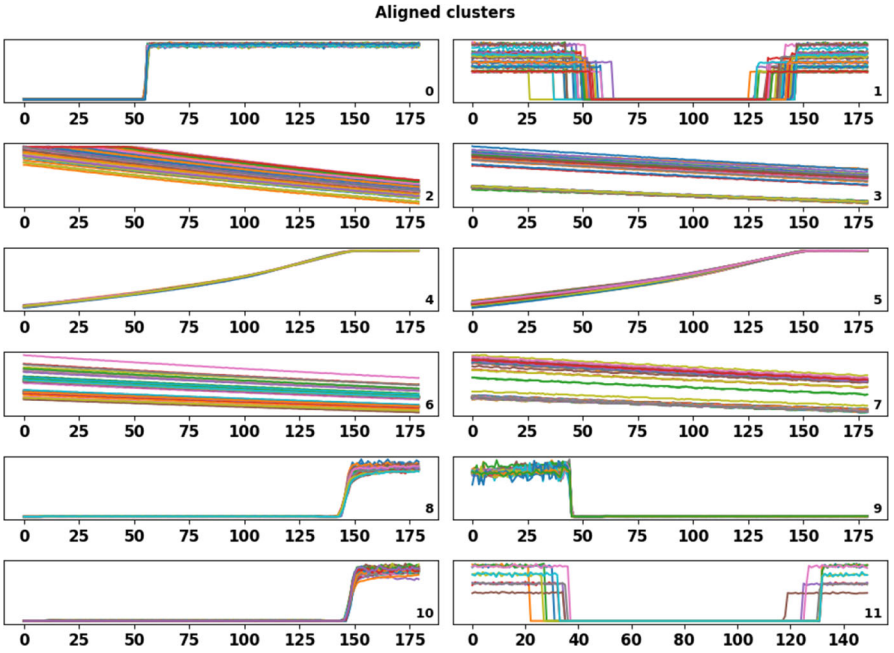

Fig. 5 The aligned clusters from Fig. 4

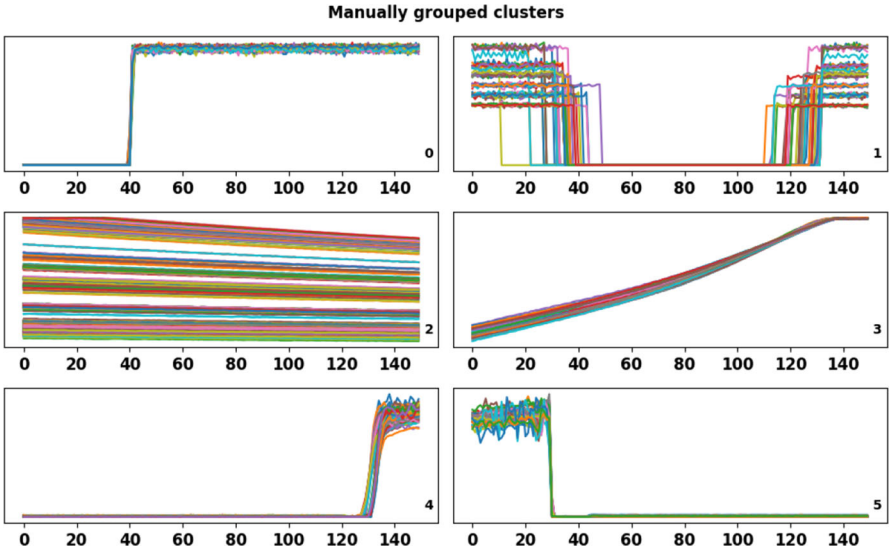

Fig. 6 One of the last steps is to manually merge the clusters from the previous agglomerative hierarchical clustering. In this figure, clusters 1 and 11 from Fig. 5 form cluster 1, clusters 2, 3, 6 and 7 are grouped into cluster 2, clusters 4 and 5 are grouped into cluster 3 and clusters 8 and 10 are grouped into cluster 4. Clusters 0 and 9 in Fig. 5 are the same as cluster 0 and 5 in this figure

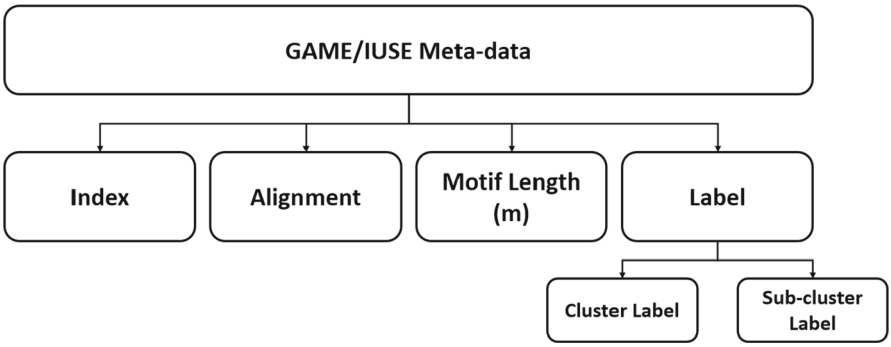

Fig. 7 A graph representation of the metadata structure produced by the GAME + IUSE processes

quick enough to be used interactively. Once the tree is pruned at a point that yields few and arbitrarily homogeneous clusters, the user has a number of available operations such as:

- Splitting and merging clusters.
- Rearranging patterns within a cluster into different sub-clusters.
- Adjusting pattern length on the group and sub-group level.
- Merging overlapping motifs.
- Shifting the starting indices of patterns on the group and sub-group level.
- Realigning patterns to minimize the distance between patterns in a group.
- Dropping clusters, sub-clusters, or individual patterns.
- Merging sub-clusters.
- Assigning sub-clusters to different parent clusters.
- Reclassifying sub-clusters into parent clusters.
- Finding the patterns within a group that deviate the most from the others in the same group.

Using these operations, the user can go from what is presented in Fig. 4 to Fig. 5 and further to Fig. 6. The introduction of sub-clusters allows the user to label variations within the same cluster using the same set of tools. Sub-clusters enable the user to track deviations, changes, and other structures within a single cluster.

4.5 A semantic data structure

The result from the previous operations is a metadata structure, shown in Fig. 7. The data structure might not seem very efficient, but the memory consumption has proven to be very small since only the interesting subsequences are annotated. The memory use of this additional data structure has empirically been found to be less than 0.5% of the memory use of the corresponding MP.

With this data structure, it is possible for the user to gain a fair understanding of the fundamental structure of the studied time series. This data structure not only provides the user with the most pervasive patterns, where the patterns are, and a partitioning of them into semantic groups, it can also be used in further mining efforts to reveal

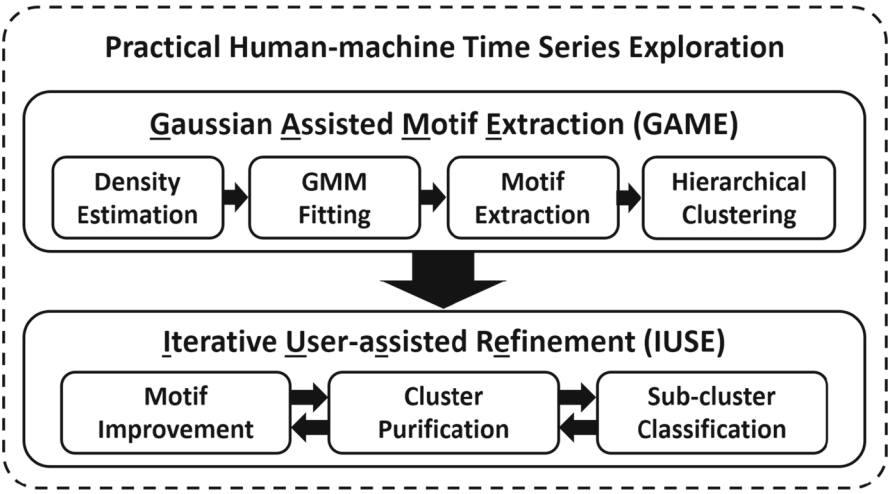

Fig. 8 A block representation of the proposed method. The two main parts GAME and IUSE are described under Sects. 4.3 and 4.4. GAME operates on top of a Matrix Profile, and IUSE further refines the clusters from GAME

more about the data. These efforts can be expected to be quite fast considering the significant reduction in data set size. A few examples illustrate this in Sect. 5.

4.6 Summary

To summarize, the proposed approach consists of two main parts illustrated in Fig. 8. The first stage (GAME) involves fitting a Gaussian Mixture Model (GMM) to the Matrix Profile values. The GMM helps separate the Matrix Profile into logical groups from which motifs (or discords) can be extracted without the need for a threshold. The motifs are clustered and presented to the user. The second phase (IUSE) involves user interaction where the user can adjust the size and alignment of the motifs, further purify clusters, and divide clusters into sub-clusters using the operations listed above.

5 Results on real-world industrial data

In this section the proposed method is demonstrated on the industrial time series data described in Sect. 3. Here it is shown that the IUSE method is practical for time series exploration, i.e. it is fast, that it can be used to segment and semantically label an unknown time series, and that it results in clusters that are homogeneous, complete, and carry semantically meaningful information.

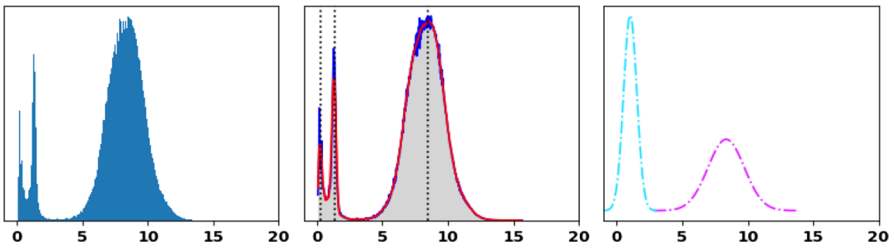

Fig. 9 The histogram of the Matrix Profile values (left) and the persistent peaks that were found (center). The best-fitting Gaussian mixture according to AIC consisted of two Gaussians (right). In the center graph, grey outlined with blue is the original distribution, red is the smoothed curve, and the peaks are marked with the dotted black vertical lines. In the right graph, darker magenta signifies a larger weight

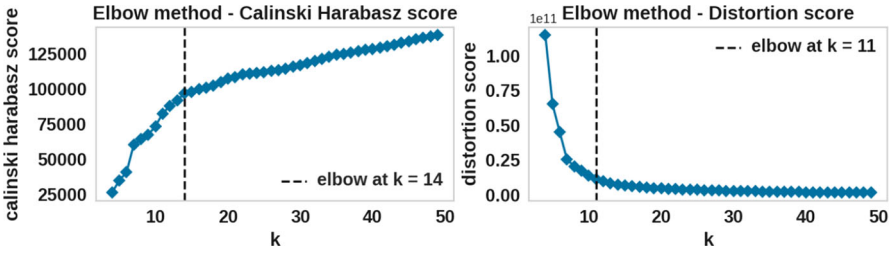

Fig. 10 The distortion measure and Calinsky-Harabasz score for 4 to 50 clusters of the motifs extracted from the Oil Purification data set. The point of maximum curvature (the elbow) is located at 11 and 14 on the x-axis

5.1 The oil purification data set

5.1.1 Finding the clusters

The results from applying the GAME step on the OP data set are shown in Fig. 9. In the center graph, gray outlined by blue, is the original histogram, and the red line is the Savitsky-Golay smoothed curve. The most persistent peaks are marked with dotted black vertical lines. In the right plot, magenta signifies a larger weight (i.e. more points). Three significant peaks are found, but the best fitting GMM using AIC has two Gaussians, and the motifs are found in the leftmost Gaussian.

The motifs are extracted and grouped into 12 clusters (see Fig. 11). This initial number of clusters was determined by using the elbow method described by Satopää et al. (2011) together with two clustering score measures: the standard distortion measure, and the Calinsky-Harabasz measure (Calinsky and Harabasz 1974). This suggested that the optimal number of clusters lies between 11 and 14, see Fig. 10. All numbers of clusters between 11 and 14 were then tried and 12 resulted in clusters with good consistency within the clusters. Clustering the relatively small number (thousands) of motifs extracted from the data set is a quick operation and can be done repeatably until the user is satisfied with it as a starting point for IUSE.

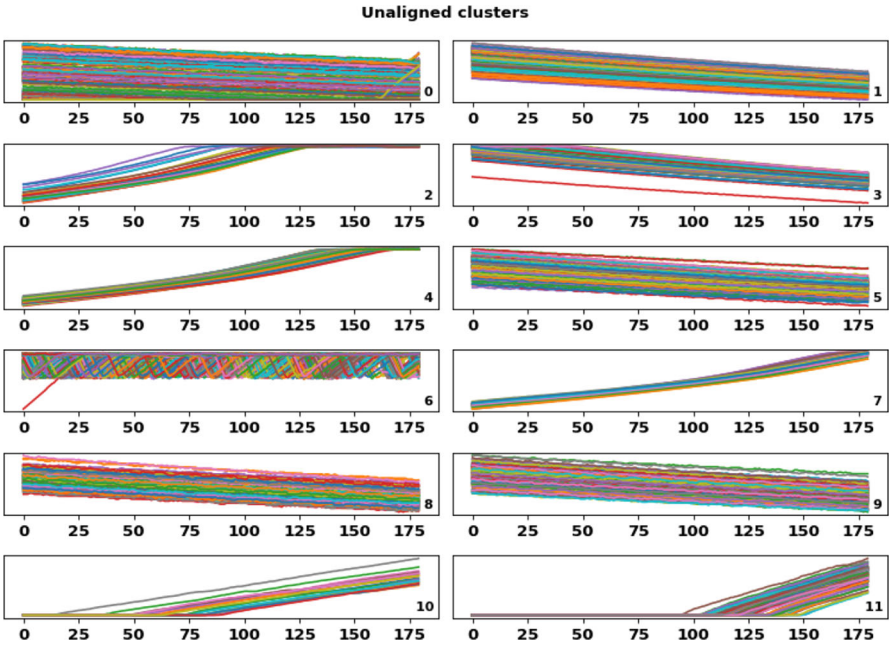

Fig. 11 The result of the hierarchical motif clustering on the Oil Purification data set

Once aligned, the clusters look as shown in Fig. 12. The number of clusters is then further reduced using the IUSE step, which gives three homogeneous clusters (denoted by the letters A, B, and C) shown in Fig. 13.

It is possible to refine the clusters further. In Fig. 13, cluster C contains two types of motifs that are very similar to each other. The first type has just one significant minimum within the window length, while the second type has more than one. These could either be considered to belong to the same semantic group or be split into two separate groups. For this illustration, they are considered to be variations of the same motifs and split into sub-clusters. The patterns are still members of cluster C, but their new labels are denoted as $C0$, $C1$, $C2$ and $C3$ as shown in Fig 14. Sub-clusters $C0$ and $C1$ have more than one significant minimum point, and the difference between them is in the size of the second drop. Sub-clusters $C2$ and $C3$ only have one drop but differ in the size of the small dip after the sharp drop.

This exploration in interaction with the user does not take very much time once the MP is computed. The MPs used here were computed on a standard Intel Core i7-8650U laptop (quad-core, 4.2 GHz) with 16 GB of RAM + 4 GB swap memory. With this setup it took roughly 7 hours[5] to calculate the MP using the SCAMP algorithm (Zimmerman et al. 2019) on a time series of length $\approx 1.2 \times 10^7$. The GAME and IUSE operations that followed each took on the order of seconds and milliseconds, largely due to the small number of extracted motifs (around 3, 000). In total, between

[5] It is likely that there would be a significant speedup if the calculations had been performed on a GPU instead.

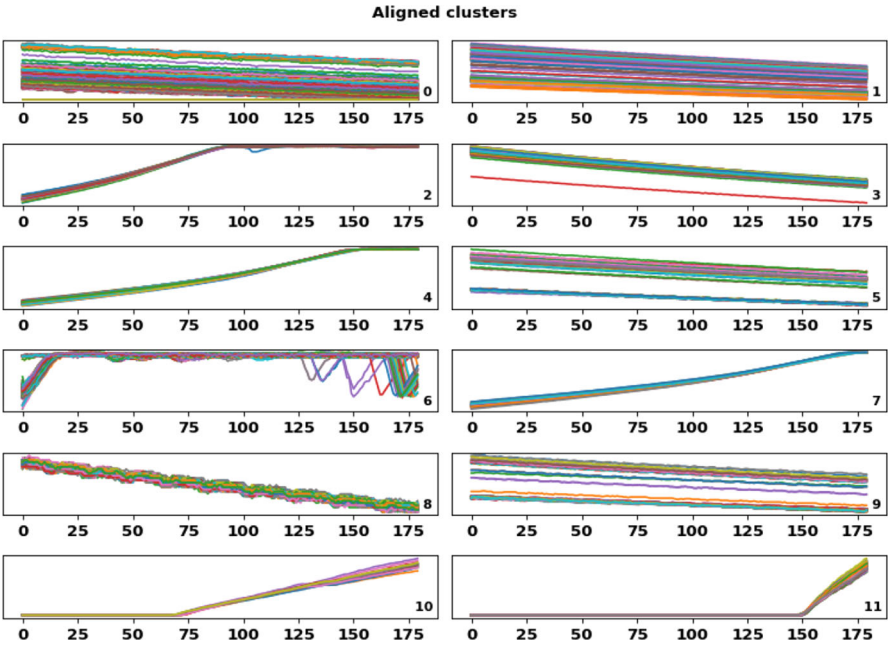

Fig. 12 The aligned clusters from Fig. 11

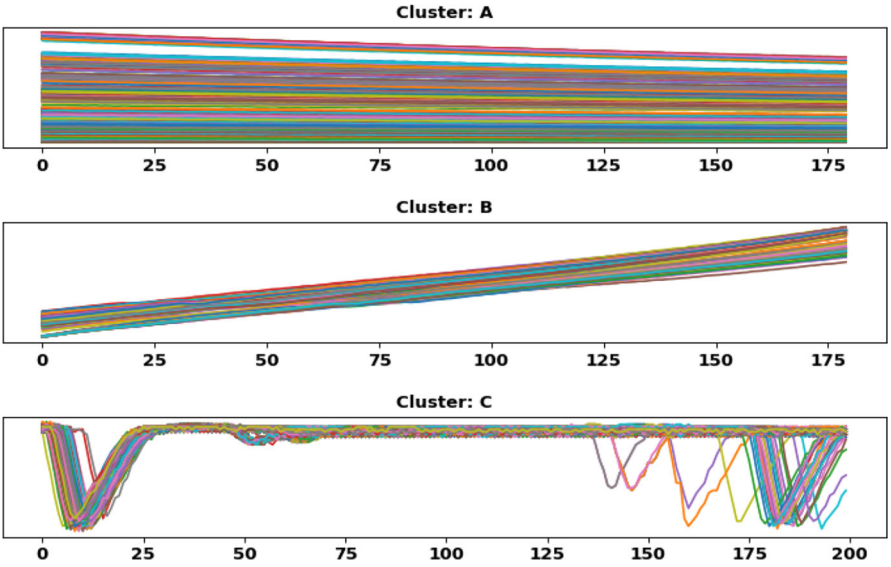

Fig. 13 One of the last steps is to merge the clusters from the previous agglomerative hierarchical clustering manually. The patterns from Fig. 12 can be refined and merged in to three groups of patterns. The three groups are labeled *A*, *B*, and *C*

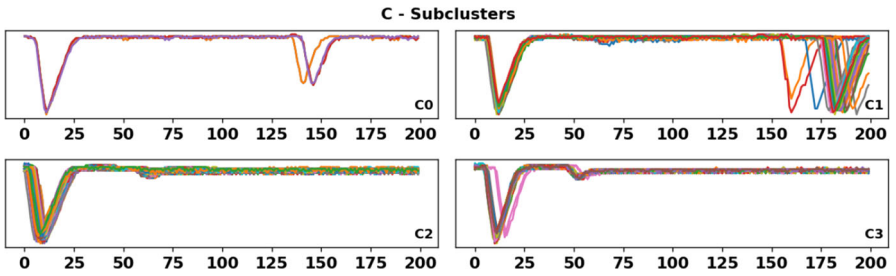

Fig. 14 Cluster C from Fig. 13 partitioned into four sub-clusters. The patterns in sub-clusters $C0$ and $C1$ have two significant minimum point, and the difference between patterns in $C0$ and $C1$ is in the size of the second drop. Sub-clusters $C2$ and $C3$ contain patterns with only one significant minimum and the difference is the size of the small dip after the large drop

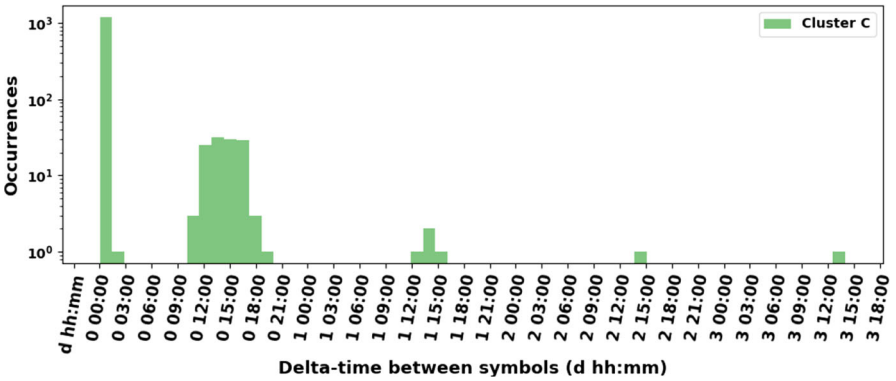

Fig. 15 A histogram of the time between repeated motifs in cluster C from a time series in the Oil Purification data set. Note that the scale on the y-axis is logarithmic. Motifs in cluster C repeat most frequently with an interval of less than 90 minutes. Cluster C is the same as in Fig. 13

45 minutes to one hour were spent performing the GAME and IUSE steps, going from an unknown time series to what is displayed in Fig. 13 and Fig. 14. Note that this is time spent by the user "exploring" the time series, not active CPU time.

5.1.2 Using the IUSE semantic data structure

The result from the GAME and IUSE steps is a metadata structure that annotates the time series with the starting index of the extracted motifs, motif lengths, cluster labels, and sub-cluster labels (see Fig. 7). This data structure gives the user a fair understanding of the fundamental structure of the time series. It provides the most pervasive patterns, where they are, and partitions them into semantic groups.

The semantic data structure can be used to derive how often and where motifs belonging to a cluster occur in a time series. Figures 15 and 16 show the intervals at which the motifs in cluster C from Fig. 13 repeat. Most motifs repeat with a time interval of one hour. Then there are both shorter and longer periods in between these motifs. The longer periods correlate to when the machine is shut down for maintenance.

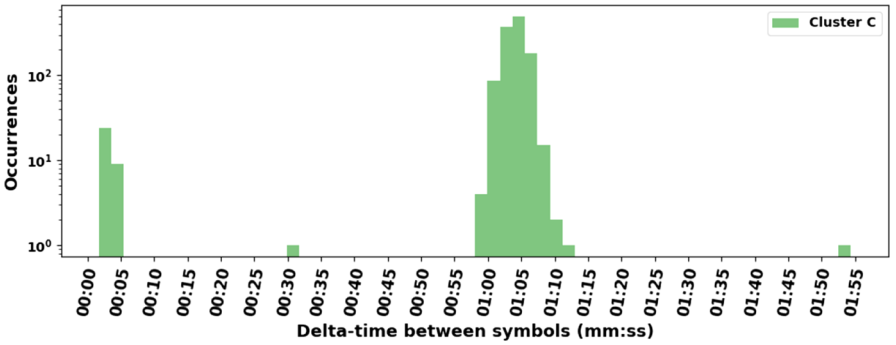

Fig. 16 A close up of the first 120 minutes of Fig. 15. The predominant interval between two consecutive motifs belonging to cluster *C* is about 65 minutes. Note that the scale on the y-axis is logarithmic and that the binning is different from Fig. 15

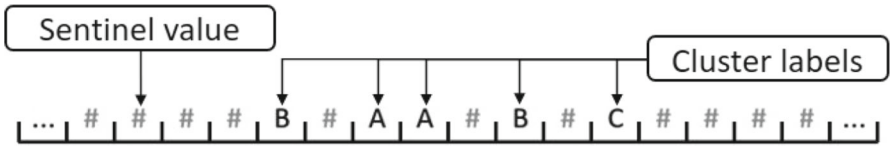

Fig. 17 A sequence containing assigned and unassigned values. The unassigned samples are marked using a special symbol, which in this case is #

The cluster and sub-cluster labels can also be used to turn the time series, or sections of it, into sequences of character symbols, which allows text mining methods to be used to mine the data. Such character sequences can be created in at least two ways. One is to create a vector of sentinel values[6] with the same length as the original time series and mark the indices representing motifs with the corresponding cluster label as shown in Fig. 17, and mark segments that do not belong to any cluster with a dummy variable.

Another way is to create a vector of only cluster labels and ignore the unlabeled segments. Figures 18 to 20 show these two vectors. Figure 18 only includes the assigned symbols (cluster labels), whereas Fig. 20 includes both, and Fig. 19 shows a closeup of Fig. 20. Each blue bar represents a motif belonging to a class. The unassigned points are labeled as "#", and the yellow dotted line in Figs. 19 and 20 is the same as the yellow dotted line in Fig. 18. The figures show that there is a repeating pattern in this data, with a small variation in the number of consecutive motifs of the same class. This is visible in Figs. 21 and 22, which show a section of the time series with the found motifs marked with colored bars. The symbols *A, B, C* correspond to the colors *red, green,* and *cyan*, respectively. The several *green* and *red* lines at the beginning and the end of a cycle, respectively, originate from the fact that the MP window length is shorter than the slopes. This means that they could be used to estimate the length of the rising and falling segments.

[6] This is a slight misuse of the phrase sentinel value. In this particular case, it means a specific flag value or dummy value signifying unlabeled data.

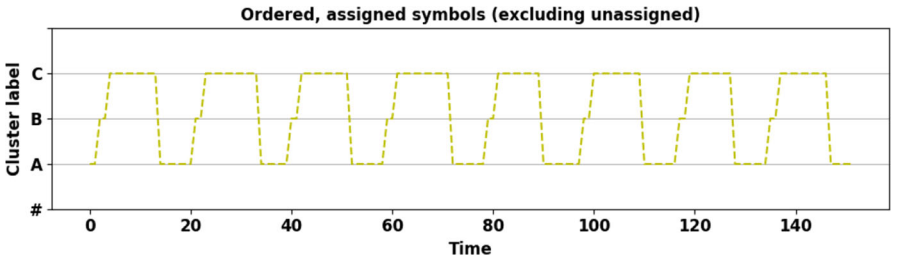

Fig. 18 The cluster labels from the new semantic data structure representing the motifs from Fig. 13

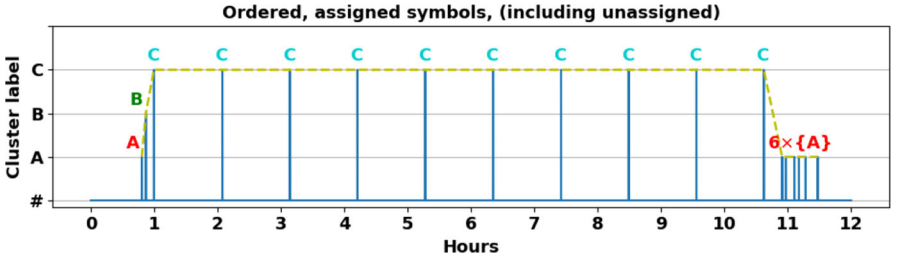

Fig. 19 A closeup of Fig. 20. The blue bars correspond to points that have been assigned symbols (labeled) (Color figure online)

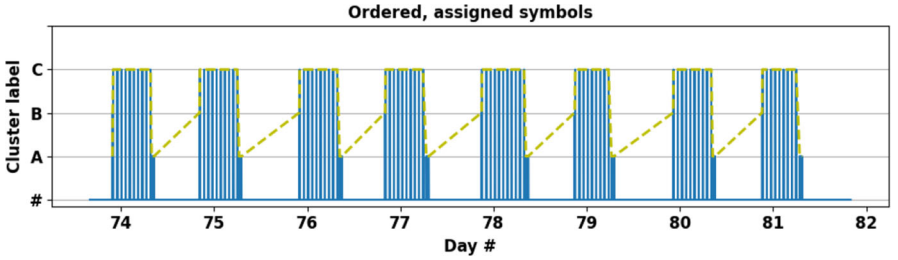

Fig. 20 The same sequence of cluster labels as in Fig. 18 (yellow) overlaid on top of a sequence that includes the unlabeled and labeled points (blue) (Color figure online)

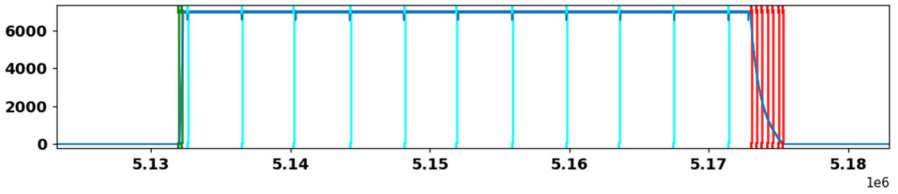

Fig. 21 A sub-section of the time series representing one process cycle that has been segmented using the motif clusters

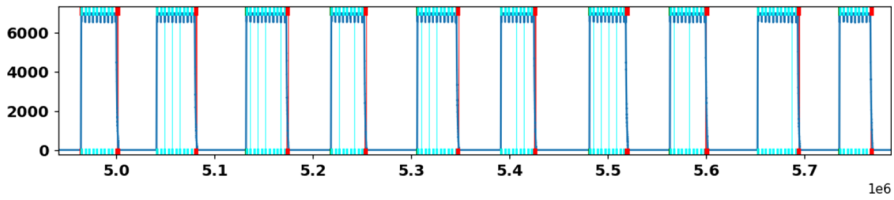

Fig. 22 A sub-section of the time series that has been segmented using the motif clusters

5.1.3 Detecting and labelling process cycles

The OP data set contains nine different signals and the motif mining results for one of them are described above. The proposed method with GAME and IUSE was applied to two more signals in the data set with very similar results. This is illustrated in Fig. 23, which shows how the motif symbols from the three time series are aligned in time. The motifs in the first signal can be categorized into one of three categories (A to C) but it can be up to seven categories (A to G) in the two remaining signals. The labels are grouped in time with longer sections of unlabeled data in between, forming cycles. This agrees with the observed behavior of the process. Figure 23 show eight of these cycles, but there are a total of 130 cycles in each signal. The number of occurrences of each unique cycle has been counted. The cycles that appeared more than once are seen as common, while a cycle that only occurs once is seen as uncommon. The common cycles from signal one and their counts are listed in Table 1. The uncommon cycles and their Levenshtein edit distance, d, (Hyyrö 2001) to the most similar cycle are listed in Table 2. All common cycles have a distance of 1 to their non-trivial closest neighbor. As shown by their nearest neighbor index (column labeled with *nn* in Table 1 and Table 2), most cycles either belong to or are neighbors to one of the top four most common cycle types. Some cycles, like cycle 16, could be considered to have a distance of 2 to the four most common cycles. But similar results were achieved without this added complexity; all cycles were therefore evaluated using only the distance to their nearest neighbor.

Among the uncommon cycles listed in Table 2 one stands out; cycle 18 has an edit distance of 12 to its closest neighbor, and it is the only cycle to have a nearest neighbor that is also an uncommon cycle. Cycle 18 is shown in the left graph of Fig. 24. Signals two and three have been analyzed using the same methodology, and the deviating cycle appears across all three.

In the remaining two signals, more than one sequence stand out from their neighbors in terms of edit distance. In the second signal there is one additional deviating cycle, but this cycle is not marked as abnormal in the other two signals. The third signal has four deviating cycles where one is the same as in the first two signals. The three remaining cycles were found to be the result of a discrepancy in the sampling and the partitioning of the data.

In summary, in the OP data set, the common cycles are discovered and shown. One significant deviating cycle is found across all three signals. Whether this and the second deviating cycle from the second signal actually is an indication of abnormal behavior is ultimately a question for an expert, but the first one can, with relative confidence, be

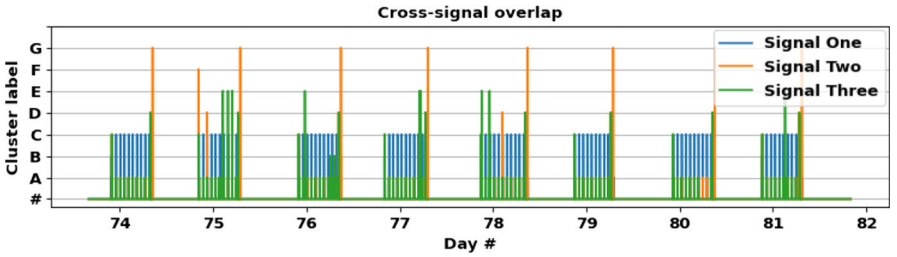

Fig. 23 The figure displays eight cycles of symbols from the three signals in the oil data set. The time scale on the x-axis is exactly the same as in the original time series. The symbols show a clear overlap which indicates a strong correlation between symbols from the three signals. Signal two is mostly obscured due to its similarity to signal three

Table 1 A table of the cycles that appear more than once in the first time series of the oil data and their counts

Cycles that occur more than once			
nr	Type sequence	Count	nn
1	[# B B C C C C C C C C C C A A A A A A A #]	25	2
2	[# B B C C C C C C C C C C C A A A A A A A #]	21	1
3	[# B B C C C C C C C C C A A A A A A A #]	17	1
4	[# B B C C C C C C C C C A A A A A A #]	12	1
5	[# A B C C C C C C C C C A A A A A A A #]	8	1
6	[# B B C C C C C C C C C C A A A A A A #]	6	3
7	[# B B C C C C C C C C A A A A A A #]	5	1
8	[# B B C C C C C C C C C A A A A A A A A #]	5	2
9	[# A B C C C C C C C C C C A A A A A A A #]	4	2
10	[# B B C C C C C C C C A A A A A A A A #]	2	1
11	[# B B C C C C C C C A A A A A A A #]	2	3
12	[# A B C C C C C C C C A A A A A A A A #]	2	5
13	[# A B C C C C C C C C C A A A A A A #]	2	4
14	[# B B C C C C C C C A A A A A A #]	2	7
15	[# A B C C C C C C C C C C A A A A A A #]	2	5
16	[# B B C C C C C C C C C C C C A A A A A A #]	2	10
17	[# B B C C C C C C C C C C C A A A A A A #]	2	2

All cycles have a distance of 1 to their nearest nontrivial neighbor (the index of the nearest neighbor is shown in the column marked "nn"). The column marked "count" shows how often the sequence appears

labeled as abnormal. In an attempt at interpreting this deviation, we had interactions with an engineer from the company that provided the data. The conclusion was that this deviation is likely not related to "*belt slippage*" or "*sludge build-up* problems. Still, it can range from "*something blocking the discharge of the dirt separated from the oil*", or "*someone who pushed the emergency stop button for some reason*", to a "*possible problem with the sensor(s)*".

Table 2 A table of the cycles that only appear once in the first time series of the oil data. Cycle 18 deviates significantly from the others with an edit distance of 12 to its nearest neighbor. The column marked "d" shows the edit distance to the nearest neighbor

Cycles that only occur once			
nr	Type sequence	d	nn
18	[# C C C A A A A A A A A B B C C C C C C C C C …C C C C C C C C C C C C C C C A A A A A A A A #]	12	23
19	[# A B C C C C C C C C C A A A A A A A #]	1	15
20	[# A B C C C C C C C C C A A A A A A A A #]	1	3
21	[# A B C C C C C C C C C A A A A A A A A #]	1	12
22	ł[# A B C C C C C C C C C A A A A A A A A #]	1	5
23	[# B B C C C C C C C C C C C C C A A A A A A A A #]	1	10
24	[# A B C C C C C C C C C C C C C A A A A A A A A #]	1	9

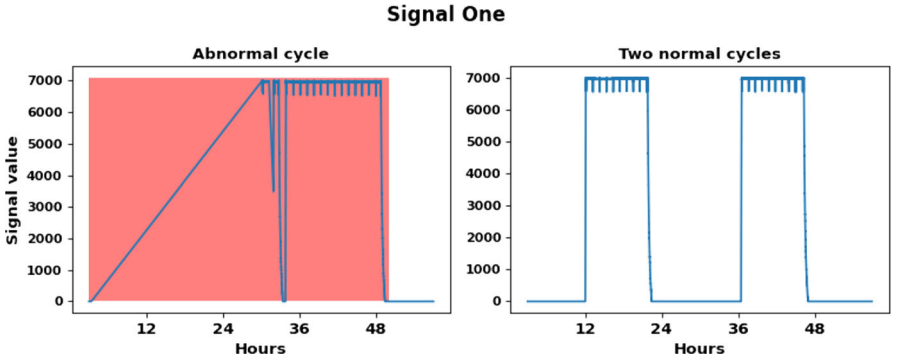

Fig. 24 An abnormal cycle, marked in red (left panel) and two normal sequences (right panel) from the first signal in the oil data set. The abnormal cycle has counterparts in the other two signals as well

5.2 The Water Treatment data set

The WT data were collected during the transitional phases between states in the water treatment process. The data segments are labeled as one of three transitions between states *"2:2"*, *"2:3"* and *"3:2"*. The GAME step resulted in 4, 386 patterns. Initial clustering, in the same fashion as described above, and further refinement with IUSE resulted in 42 clusters with reasonable perceived similarities. A selection of the clusters is shown in Fig. 25.

Comparing the cluster assignments to the true labels in the data set showed that 37 out of 42 clusters were pure, i.e., only contained patterns belonging to one state transition, and about 92% of all patterns were assigned into pure clusters. Two types of experiments were done to evaluate whether this is a good result, i.e., to see if the proposed method resulted in a segmentation and clustering that reflected the true state transitions. The first is an experiment where different ways to do the segmentation and the clustering are used to see if a purer grouping can be achieved. The second is a statistical hypothesis test to see if the results are unlikely to occur by random chance.

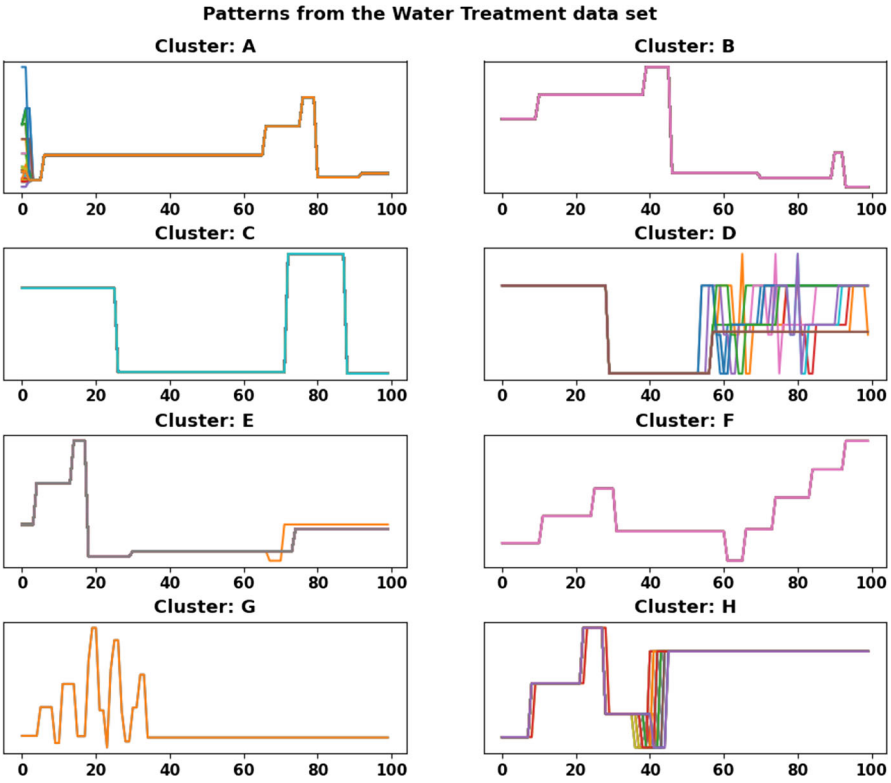

Fig. 25 Eight out of the 42 clusters from the Water Treatment data set. The patterns in these clusters have a high similarity. That is also true for the majority of the remaining clusters

5.2.1 Comparing clustering and segmentation results

The goal with this experiment is to check if the proposed method provides a better segmentation into subsequences and subsequent clustering into semantic groups than three alternative approaches described below.

1. GAME without human intervention (i.e. no IUSE). Here the subsequences resulting from GAME are subject to hierarchical clustering without human interaction. This test is done to measure the effect of IUSE.
2. Using perfect information about state transitions. Longer sequences are first extracted from the time series using the state labels. The sequences are then further split into subsequences of length 100. The subsequences are then subject to hierarchical clustering without human intervention. The assumption is that this produces subsequences that cluster together into semantically pure clusters.
3. Segmenting the time series without any information. The time series is simply split from beginning to end into subsequences of length 100. The subsequences are then subject to hierarchical clustering without human intervention. The assumption is

that this should produce subsequences that do not cluster together into semantically pure clusters.

In all three cases, several break points are used in the clustering, resulting in different numbers of clusters.

The aspect of "better" or "worse" clustering is measured with homogeneity and completeness (Rosenberg and Hirschberg 2007). Homogeneity is the measure of how pure the clusters are. Maximum homogeneity can be achieved by assigning each element into its own cluster. Completeness is the measure of how few (or many) clusters are needed to cover all patterns, and maximum completeness can be obtained by assigning all elements to one big cluster. Usually it is difficult to achieve high homogeneity and high completeness at the same time; there is a trade-off between them. It is desirable to achieve as high homogeneity as possible for a given completeness value.

The homogeneity and completeness are shown in Figs. 26 and 27. The proposed method (GAME followed by IUSE) achieves a higher homogeneity for the corresponding completeness than the three comparison methods. Looking at the homogeneity and completeness scores individually in Fig. 26 shows that the proposed method beats the three baseline methods in both scores. Extracting patterns assuming perfect information is the second runner up in homogeneity score but performs the worst in completeness. However, the difference in completeness between the methods is quite small. Levels of homogeneity equal to that of the proposed method are achieved around 100 clusters when perfect information is assumed, around 330 clusters when solely using GAME and an equal level of homogeneity is never reached when no information is assumed. In terms of completeness, all the baseline methods are surpassed with a good margin.

5.2.2 Statistical evaluation

The results from the segmentation and clustering experiment above imply that there is a connection between the labels and the motifs. However, the data set is very imbalanced and the result could come about purely as an effect of this imbalance. The goal of this subsection is to investigate whether this is the case or not. To do this a hypothesis test was set up, with the null hypothesis that there is no connection between the semantics in the transitions and the motifs. The test was carried out by computing the probability of getting a result that is as good or better if the null hypothesis is true with the same data set imbalance, and reject the null hypothesis if this probability is very low.

The total number of subsequences extracted from the water treatment time series is $4,386$; where $4,273$ (97.4%) belong to the transition "2:2", 102 (2.3%) belong to the transition "2:3", and 11 (0.25%) belong to the transition "3:2". The proposed method (GAME followed by IUSE) distributes the $4,386$ subsequences among 42 clusters, of which 37 are pure and 5 are mixed (i.e. contain patterns from more than one transition). A total of $4,294$ motifs (97.9%) are assigned to pure clusters while 92 (2.1%) are assigned into mixed clusters

The test is done in a Monte Carlo fashion by using $4,386$ "balls" colored into three categories with the above distribution, sorting them one million times into 42 "boxes",

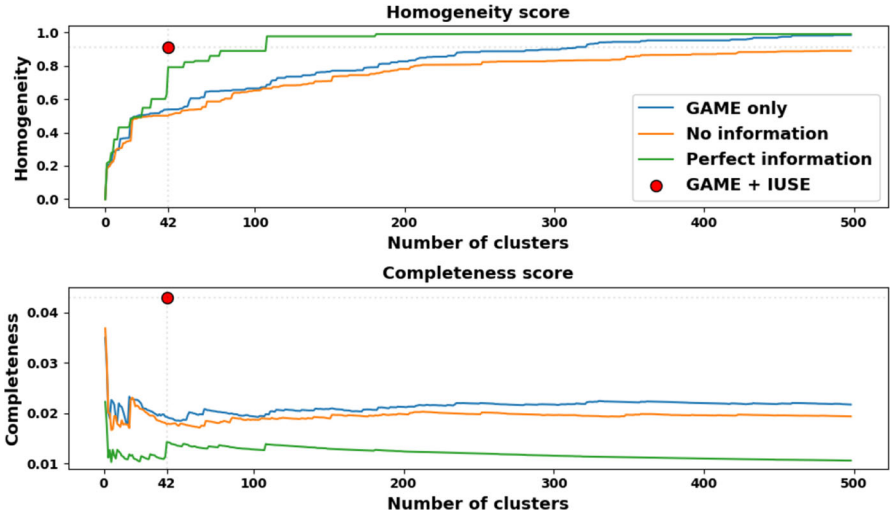

Fig. 26 The homogeneity and completeness scores for the three baseline methods as well as the proposed method. The human user through IUSE outperforms the three baseline methods both in terms of completeness and homogeneity. The special case with one cluster and $Completeness = 1$ is not shown in the figure

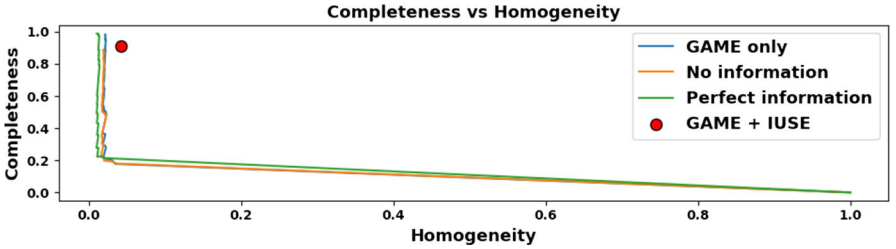

Fig. 27 GAME in combination with IUSE scores higher in both homogeneity and completeness for all values

with at least one "ball" in each box, and computing how often this sorting results in at least 37 of the "boxes" being pure (i.e. containing "balls" of only one color), or that at least 4, 294 of the "balls' are assigned to pure clusters. The outcome of the test is that it never happens, in one million trials. Thus, the p-value is less than 10^{-6}. In one million random trials, the highest observed purity was 24.6% (i.e 1, 079 patterns were assigned into pure clusters). It was only in 15 out of the one million trials that more than 1, 000 patterns were placed in pure clusters. It is thus safe to reject the null hypothesis and accept that there is a connection between the semantics and the motifs found.

Furthermore, the connection between the clustering and the semantics seems to be strong. Figure 28 shows that getting almost 98% of the patterns sorted into pure clusters, with only 42 pure clusters, is very unlikely to happen by chance. It is not until the number of clusters reaches into the thousands that the likelihood for getting almost 98% in pure clusters reaches credible levels. Each set of green, yellow and red

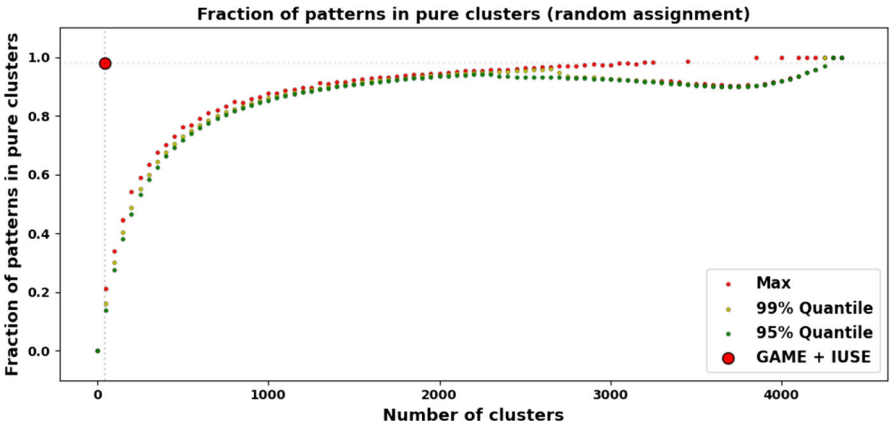

Fig. 28 Each set of green, yellow and red dot represent 10,000 random allocations to x clusters (the number of clusters are shown on the x-axis). The plot illustrates that it is very unlikely to reach close to 98% of patterns in pure clusters if the numbers of clusters is much lower than 2,000 (Color figure online)

dots in Fig. 28 represents 10, 000 random allocations. Green, yellow and red signifies the 95^{th} and 99^{th} quantile as well as the maximal fraction of patterns in pure clusters.

5.3 The City Bus Fleet data set

Two problems occurred in the bus data. The first is irregular sampling. Data samples come at varying intervals of 1-2 seconds, but there are many periods with missing values, some of which last up to 30 seconds. These dropouts caused many motifs to be gathered around the sharp spikes created by the missing values. The sampling issue could be solved by resampling the time series and removing sections with longer dropouts. The second problem is highly varying pattern lengths. As shown in Fig. 29, the bus data consists of a sawtooth-like pattern where the length of the falling edge varies greatly. This is normal since the data represent the air pressure, which is not consumed at a fixed rate. This variation made it challenging to select the window length when calculating the MP. Matching the window length to the shorter patterns resulted in the inability to catch the longer patterns properly, and matching the window length to the longer periods resulted in many of the shorter patterns appearing within one window. Finding one suitable length proved impossible.

Testing a dynamic or flexible time window for the MP computation was beyond the scope of this paper. However, one way to deal with this problem could be the solution presented by Madrid et al. (2019), which has a time complexity of $O(n^2 r)$ since it calculates one MP for each length in a range r. Another way could be to use a Dynamic Time Warping (DTW) based algorithm like SWAMP, which is slower to compute than Euclidean distance based algorithms such as SCRIMP++ and SCAMP.

Another observation on the CBF data is that there is no distinct group of motifs that is separate from the rest of the time series segments. The MP of the air pressure data from the buses has a basically unimodal right-skewed distribution like in Fig. 30. This is expected for a time series that is periodic but with varying period lengths. Most

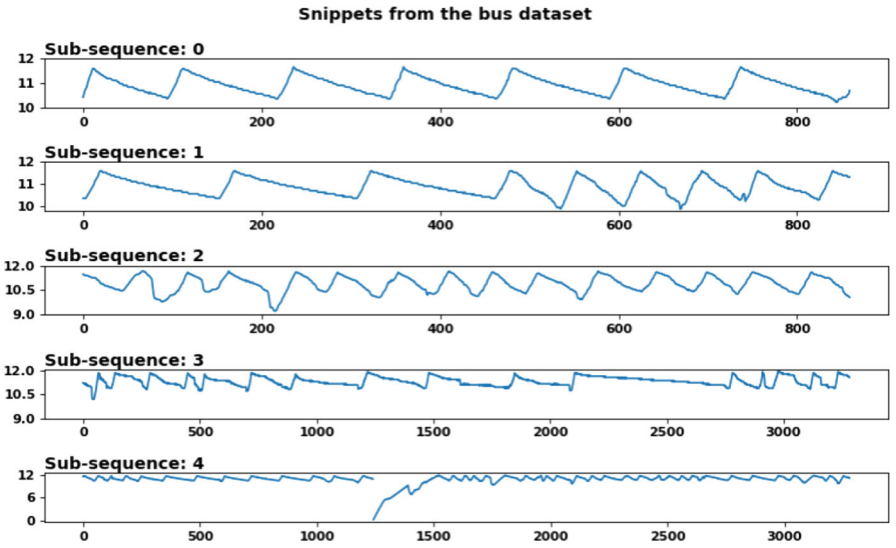

Fig. 29 Five subsequences pulled from the bus data set. The sawtooth-like pattern present in all the subsequences varies significantly in length. This causes trouble for lockstep distance metrics like Euclidean distance (note that the last two subsequences are longer)

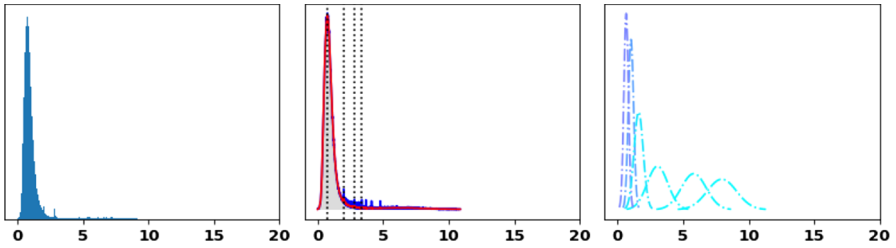

Fig. 30 The distribution and GMM of the bus data using the same format as Fig. 9. The distribution of the Matrix Profile (left), the persistent peaks that were found (center) and the best Gaussian mix (Right)

of the subsequences had a medium to low distance to the closest non-trivial match and there are only a few discords in the distribution's right tail. The discords were primarily significant drops of pressure, like in subsequence 4 in Fig. 29. These drops in pressure likely correspond to draining the wet tank, which typically happens in the beginning or end of a work shift.

There were, despite the time warping issue, some interesting findings on the city bus data. The buses had reached an operational life of about five years, and this meant that some of the compressors started to perform poorly and were replaced. On two out of three buses that had a compressor replacement, it is possible to see a change in the frequency of motifs belonging to different clusters, see Fig. 31. The change can be seen across the different clusters and happens in conjunction with the compressor change. The frequency score was calculated by dividing the number of motifs by the total number of samples on a week per week basis. The vertical lines represent workshop visits where the line in the center signifies a compressor replacement. After

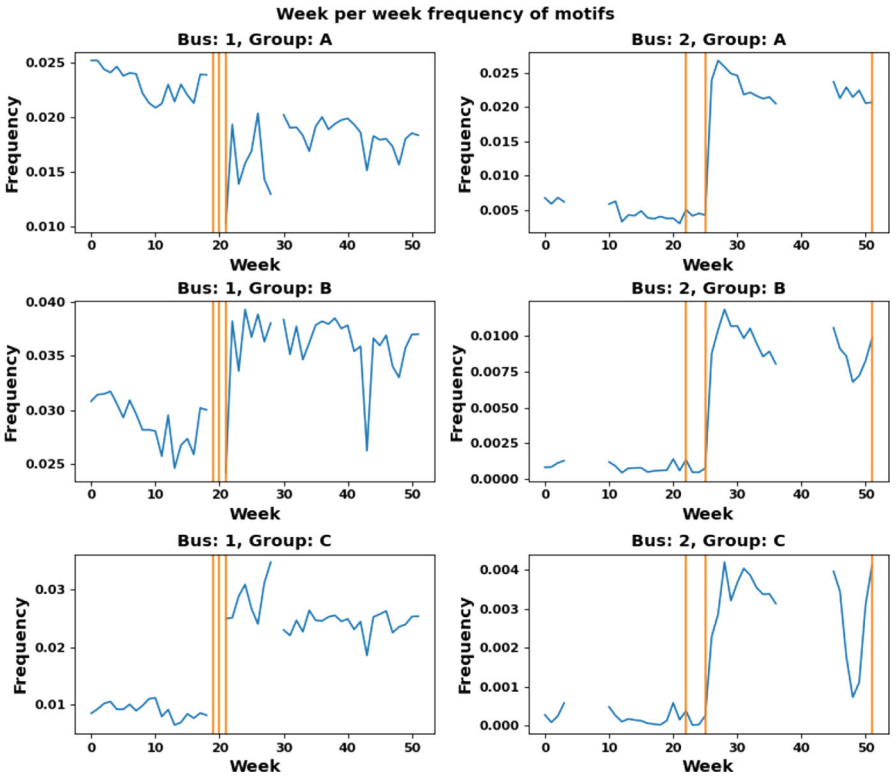

Fig. 31 The plot shows the frequency of motifs in a motif group over time. The vertical lines indicate workshop visits. Bus 1 had its compressor replaced during week 20 and bus 2 had a compressor replacement during week 25

each replacement, there is a persistent change in the motif frequency of the motifs in each group. There were six groups per bus and the change could be seen across all of the groups. While this is far from detecting or predicting a failure, it indicates that there could be a correlation between motif and difference in behavior that a new compressor would cause.

Table 3 summarizes, for each dataset: the motif length used, how many peaks were found, and how many clusters were found after using GAME and after human refinement with IUSE. The number of clusters found after using GAME, which is the result of an hierarchical clustering, does not really affect how many clusters are left after the human refines them with IUSE. The number of GAME clusters is a subjective choice that is just a starting point for IUSE.

6 Benchmark results on ECG data

The same approach as described in the previous section for the industrial time series was applied to the ECG Arrythmia time series. A window size of 192 was used to

Table 3 Summary of the subsequence length used, the number of peaks found, and the number of motif clusters before and after human refinement with IUSE for each dataset

Dataset	Subsequence length	# Peaks	# Clusters after GAME	# Clusters after IUSE
OP	180	3	11	3
WT	100	4	16	42
CBF	30	4	12	11

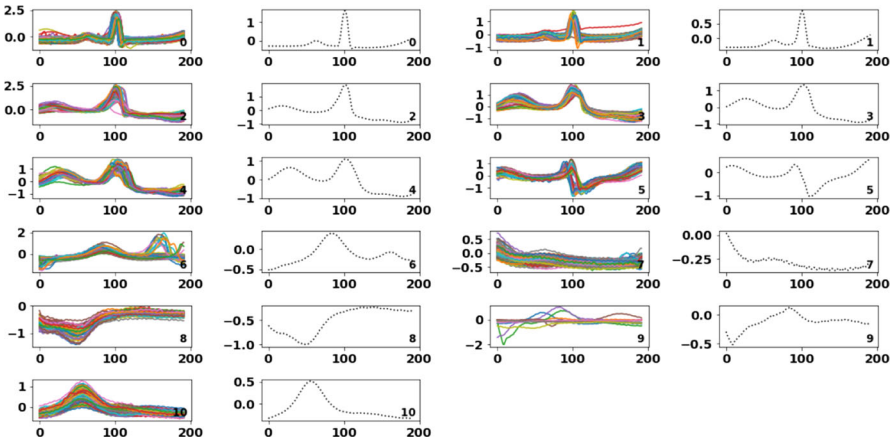

Fig. 32 The resulting clusters from applying GAME and IUSE presented in sections 4.3 to 4.4. Clusters 0 and 1 contain labeled normal beats, clusters 2 to 5 contain labeled abnormal beats (the cluster label is shown in the bottom right corner in the plots). The remaining clusters contain mostly unlabeled parts of the time series. The first and third column contain all the extracted motifs overlayed on top of each other. Columns two and four depict the average motifs in each cluster

calculate the matrix profile. The MP histogram contained one small and one large peak and was best approximated with four Gaussian components. Twenty GAME clusters generated a good starting point for IUSE, which finally resulted in the eleven clusters shown in Fig. 32.

The first clusters are quite pure, i.e. contain almost only either normal beats or premature ventricular contraction (PVC) beats. For example, the top left cluster in Fig. 32 has 593 patterns: 590 labeled as normal and 3 labeled as PVC, which corresponds to a cluster purity of 99.5% . However, when we reach later clusters, e.g. the fourth one in the left column, then many of the peaks are not labeled. This is because the sections fall between two labeled peaks, which is an issue that is not unique for our method; we see the same with, e.g., HubFinder (Yoshimura et al. 2019). We can choose to either ignore or include these "unlabeled" patterns in the evaluation, and we include them since our setting is the case when a user is exploring a data set without known labels.

Figure 33 shows how the cumulative purity changes as more and more clusters are included in the set, for the first six clusters. The x-axis is the cumulative number of beats in the included clusters. Comparing these results to Fig. 8 in Yoshimura et al. (2019) shows that the sample purity achieved with IUSE is competitive to that by

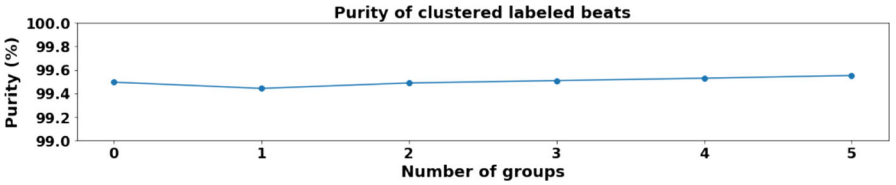

Fig. 33 The cumulative purity of clusters 0 to 5 for the ECG data set, starting at 99.5% and ending at 99.6%. Evaluation of the purity beyond six clusters is left out since most of the motifs in the remaining clusters lack labels

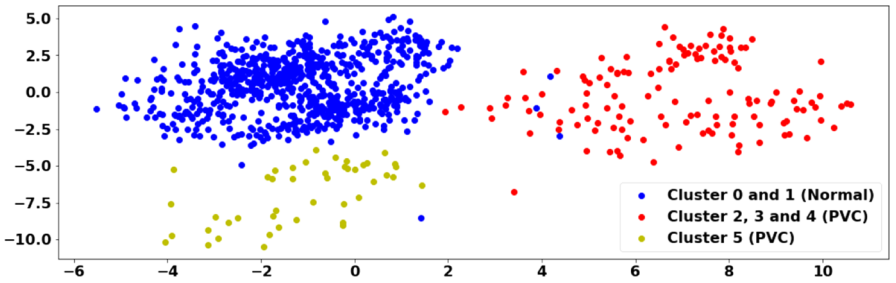

Fig. 34 The annotated heartbeats projected in 2D space using Multidimensional Scaling (MDS). The figure indicates that there are at least two types of PVC beats, which agrees with the results in Fig. 32. The PVC beats, type one and two, are represented by the red and yellow dots while normal beats are represented by the blue dots (Color figure online)

HubFinder (99.4%), which in turn is better than that of other previously proposed methods on this data.

An additional finding with IUSE is that there is more than one variant of PVC beats. The top clusters, both left and right column, in Fig. 32 correspond to normal beats, and the patterns in them are quite similar. The clusters in rows 2 and 3, both left and right columns, correspond to PVC beats. Arguably, the cluster in row 3 in the right column shows patterns with a somewhat different shape than in the other three clusters. This is also illustrated in Fig. 34, which is a multidimensional scaling projection of the beat patterns (extracted using the ground truth), where there appears to be at least two clusters of PVC beats.

7 Discussion

A trial and error based exploratory approach for setting the window size was sufficient for the OP and WT data, whereas it was impossible to find a suitable window length for the CBF data. The MP is tolerant in terms of window size and will generally conserve patterns shorter than or close to the length of the sliding window. Nevertheless, since the proposed approach operates on top of an MP, it could also operate on top of a Pan-Matrix Profile (Madrid et al. 2019) with minor modifications.

All experiments in this paper were performed using Euclidean distance-based MP algorithms. It is possible, to some degree, to handle variable length, warped and scaled

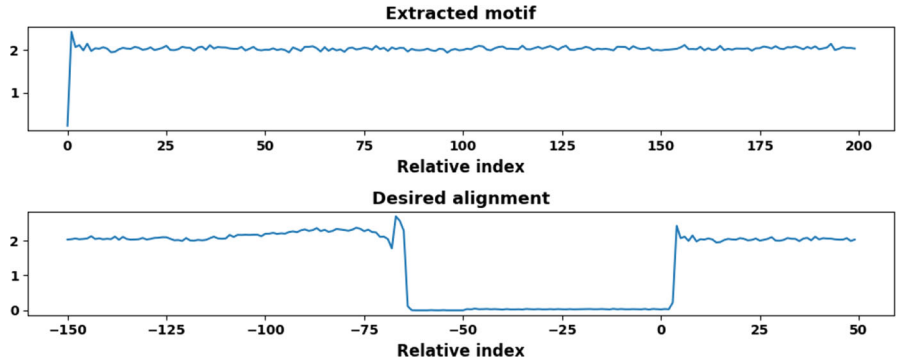

Fig. 35 It is documented that non-complex sections tend to be premiered in the Matrix Profile (Dau and Keogh 2017). This leads to alignments like in the top image that finds the sought after event but includes a disproportionate amount of the flatter sections instead of the event itself

motifs using IUSE. The MP is capable of picking up motifs that are within two-thirds of the original window size (Yeh et al. 2016) and the user is able to shift and resize motifs during the IUSE process. This results in some flexibility when the window size is selected as well as some robustness to variable length motifs. However, the results suggest that using a method based on DTW or a method capable of handling multiple length motifs would do better under circumstances with non-stationary wavelengths like the CBF data. A recent paper introduced the SWAMP algorithm (Alaee et al. 2020) where the MP is efficiently calculated using DTW. SWAMP has a worst-case time complexity of $\mathcal{O}(n^2m^2)$. This can be compared to the fast SCRIMP, which is of order $\mathcal{O}(n^2)$. Since GAME and IUSE operate "on top of" any MP, it would be possible to perform the same experiments using a DTW based algorithm like SWAMP.

Several attempts were made to use the types of annotation vectors suggested by Dau and Keogh (2017) but with little success. The main problem when Corrected Matrix Profiles (CMPs) were applied was motif alignment, both in the OP data and in the CBF data. It is documented that non-complex sections tend to be premiered in the MP (Dau and Keogh 2017). This leads to alignments issues like in the top image of Fig. 35 where the sought after event is found but the less complex areas surrounding the motif are favored, and the event is shifted to one side.

Similar procedures were attempted on the CBF data, where the goal was to align the motifs to the sawtooth pattern. It was generally complicated to create an annotation vector that produced a good alignment. Additionally, several parameters had to be tuned, which increased the difficulty. This observation does not disprove the utility of the CMP, but it substantiates the argument for using a human-machine cooperation as the one presented in this paper. Instead of using a very complex solution that may be difficult to tune, the idea is to opt for letting the machine do the repetitive, time consuming (boring) parts like motif extraction, motif discovery, and clustering, and have the human do the more complex and trickier tasks.

Similar arguments can be made about Ostinato and the Semantic Motif Finder algorithm. Attempts were made to use Ostinato to align motifs and discover motif chains but fell victim to the same simplicity bias as in Fig. 35, which increased the

complexity of the task. The Semantic Motif Finder algorithm was straightforward to use, but the added capabilities were not needed and came at the cost of a more complex solution. However, a semantic MP, produced by Semantic Motif Finder, could be used since GAME and IUSE are agnostic to the specific method used to compute the MP.

In the OP data set, the common cycles are discovered and one significant deviating sequence is found across all three signals. It is worth mentioning that a method called CycleFootPrint (Fanaee Tork et al. 2020) has been applied to an oil purification data set similar to the OP data used in this paper. The method has a worst-case time complexity of $\mathcal{O}(n^2)$. Based on the illustrations in (Fanaee Tork et al. 2020), it seems to find patterns that are similar to the motifs found by the proposed GAME/IUSE approach. However, making a quantifiable comparison would require further ground truth for the oil data. The MP algorithm, SCRIMP++, used by GAME/IUSE also has a time complexity of $\mathcal{O}(n^2)$, but can practically compute the MP very quickly since it is an anytime algorithm. It has been shown that SCRIMP++ converges fast and that only a small percentage of the calculations need to be performed to get a close approximation of the final MP (Zhu et al. 2018).

On the WT data, the proposed method manages to assign approximately 98% of all motifs into pure clusters. Does that mean that 98% of the motifs are semantically correlated and that the remaining 2% are not? During the exploration phase some motifs were observed that appeared visibly very similar but were assigned to different labels. However, the IUSE process resulted in better assignments than the case where perfect information (the labels) was used to extract the patterns. Thus, the IUSE method seems to produce very good assignments on the WT data.

Finally, there can be detectable traits that predict the need for bus compressor maintenance in the CBF data. This has been shown in (Fan et al. 2015, 2016) using recurrent neural networks. The impression is that this could possibly be done using a motif finding process with the MP calculated under dynamic time warping. At least there is a difference in motif frequencies before and after a compressor replacement (which does not say that the motifs have predictive power).

8 Conclusion

This paper presents an approach for a human expert to work with the Matrix Profile and motifs to analyze and explore industrial time series. The contribution is a virtually parameterless procedure (GAME) for extracting motifs from a Matrix Profile as well as a tool to be used by a human expert (IUSE) to enhance, cluster, and thereby extract the semantic meaning of motifs from an industrial time series. The paper also provides an overview of some general experiences when working with the Matrix Profile on real-life industrial data.

The proposed method works on top of a Matrix Profile. This is important since it allows the users to select a Matrix Profile algorithm that suits their needs. The main reason for choosing the SCAMP algorithm to calculate the Matrix Profile here was the ability to tile the input data during the calculation. The sizes of the data sets vary slightly, but all of them contain about one year of sensor readings, each from a real industrial environment. The tiling feature made it possible to perform the Matrix

Profile calculations on a standard PC, and SCAMP was the only algorithm found that could do this on the entire time series on the available hardware.

The calculations of the Matrix Profile took about seven hours per time series of 1.2×10^7 points. This should be compared to the approximate one hour it takes to properly explore the time series and group motifs into semantically meaningful clusters. The Matrix Profile calculation requires no supervision, while Iterative User-assisted Refinement (IUSE) is an interactive knowledge-building process.

The Gaussian Assisted Motif Extraction (GAME) procedure is a novel way of extracting motifs from a time series. It does not require the user to set a threshold of how many motifs to extract like get top k motifs (Algorithm 1) or state of the art methods like HubFinder (Yoshimura et al. 2019). The motifs are instead extracted from the time series based on natural formations in the distribution of the Matrix Profile and then presented to the user in groups. Variable-length and warped motifs can be accounted for with the utilities of IUSE. Still, cases like the CBF data probably require that GAME/IUSE operate on top of a DTW based or variable length capable Matrix Profile like suggested in any of (Imani and Keogh 2019; Madrid et al. 2019; Alaee et al. 2020).

IUSE offers an appealing solution to the motif-finding problem; let the machine do the tedious and boring labor-intensive, repetitive work and leave the human expert with the complex, less repetitive, and more exciting tasks. The Matrix Profile can be calculated on data sets of industrial size. A human expert can, through IUSE, find semantically meaningful structures in industrial time series data. IUSE operates on top of the Matrix Profile, and the resulting data structure can be used to answer many questions about industrial time series data, which often lack reliable ground truth. The benchmark test on the labeled ECG time series shows that the suggested GAME/IUSE results in an accurate separation of normal and abnormal patterns, as well as a detection of subpatterns, achieving a purity in line with current state of the art methods on this time series. Human-machine exploration profits from the computational power and endurance of a computer along with the intellect and understanding of the human mind.

Funding Open access funding provided by Halmstad University.

References

Akaike H (1974) A new look at the statistical model identification. IEEE Trans Autom Control 19(6):716–723. https://doi.org/10.1109/TAC.1974.1100705

Alaee S, Kamgar K, Keogh E (2020) Matrix profile XXII: Exact discovery of time series motifs under DTW. In: 2020 IEEE International Conference on Data Mining (ICDM), pp 900–905, https://doi.org/10.1109/ICDM50108.2020.00099

Bagnall A, Hills J, Lines J (2014) Finding motif sets in time series. arXiv preprint arXiv:1407.3685 https://doi.org/10.48550/ARXIV.1407.3685

Calinsky T, Harabasz J (1974) A dendrite method for cluster analysis. Commun Stat 3(1):1–27. https://doi.org/10.1080/03610927408827101

Chiu B, Keogh E, Lonardi S (2003) Probabilistic discovery of time series motifs. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. https://doi.org/10.1145/956750.956808

Dau HA, Keogh E (2017) Matrix profile V: A generic technique to incorporate domain knowledge into motif discovery. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Association for Computing Machinery, New York, NY, USA, KDD '17, p 125-134, https://doi.org/10.1145/3097983.3097993

Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J Royal Statist Soc Ser B (Methodological) 39(1):1–38

Fan Y, Nowaczyk S, Rögnvaldsson T (2015) Evaluation of self-organized approach for predicting compressor faults in a city bus fleet. In: Procedia Computer Science, Elsevier, pp 447–456

Fan Y, Nowaczyk S, Rögnvaldsson T, et al (2016) Predicting air compressor failures with echo state networks. In: Third European Conference of the Prognostics and Health Management Society 2016, Bilbao, Spain

Fan Y, Nowaczyk S, Rögnvaldsson T (2020) Transfer learning for remaining useful life prediction based on consensus self-organizing models. Reliab Eng Syst Saf 203(107):098

Fanaee Tork H, Bouguelia MR, Rahat M, et al (2020) Cyclefootprint : a fully automated method for extracting operation cycles from historical raw data of multiple sensors. In: IoT Streams for Data-Driven Predictive Maintenance and IoT, Edge, and Mobile for Embedded Machine Learning, Communications in Computer and Information Science, pp 30–44, https://doi.org/10.1007/978-3-030-66770-2_3

Freedman D, Diaconis P (1981) On the histogram as a density estimator: L2 theory. Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete 57(4):453–476

Huber S (2021) Persistent homology in data science. In: Haber P, Lampoltshammer T, Mayr M et al (eds) Data Science - Analytics and Applications. Springer Fachmedien Wiesbaden, Wiesbaden, pp 81–88

Hyyrö H (2001) Explaining and Extending the Bit-parallel Algorithm of Myers. University of Tampere, Department of Computer and Information Sciences, Julkaisusarja A

Imani S, Keogh E (2019) Matrix profile XIX: Time series semantic motifs: a new primitive for finding higher-level structure in time series. In: 2019 IEEE International Conference on Data Mining (ICDM), pp 329–338, https://doi.org/10.1109/ICDM.2019.00043

Keogh E, Lin J (2005) Clustering of time-series subsequences is meaningless: implications for previous and future research. Knowl Inf Syst 8(2):154–177

Lin J, Keogh E, Lonardi S, et al (2002) Finding motifs in time series. In: Proceedings of the Second Workshop on Temporal Data Mining, pp 53–68

Madrid F, Imani S, Mercer R, et al (2019) Matrix profile XX: Finding and visualizing time series motifs of all lengths using the matrix profile. In: 2019 IEEE International Conference on Big Knowledge (ICBK), pp 175–182, https://doi.org/10.1109/ICBK.2019.00031

Moody G, Mark R (2001) The impact of the MIT-BIH arrhythmia database. IEEE Eng Med Biol Mag 20(3):45–50. https://doi.org/10.1109/51.932724

Mueen A (2014) Time series motif discovery: dimensions and applications. WIREs Data Min Knowl Discovery 4(2):152–159. https://doi.org/10.1002/widm.1119

Nakamura T, Imamura M, Mercer R, et al (2020) Merlin: Parameter-free discovery of arbitrary length anomalies in massive time series archives. In: 2020 IEEE International Conference on Data Mining (ICDM), pp 1190–1195, https://doi.org/10.1109/ICDM50108.2020.00147

Nunthanid P, Niennattrakul V, Ratanamahatana CA (2012) Parameter-free motif discovery for time series data. In: 2012 9th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, pp 1–4, https://doi.org/10.1109/ECTICon.2012.6254126

Renard X (2017) Time series representation for classification : a motif-based approach. PhD thesis, Université Pierre et Marie Curie-Paris VI

Rosenberg A, Hirschberg J (2007) V-measure: a conditional entropy-based external cluster evaluation measure. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). Association for Com-

putational Linguistics, Prague, Czech Republic, pp 410–420, https://www.aclweb.org/anthology/D07-1043

Satopää V, Albrecht J, Irwin D, et al (2011) Finding a "kneedle" in a haystack: Detecting knee points in system behavior. In: 2011 31st International Conference on Distributed Computing Systems Workshops, pp 166–171, https://doi.org/10.1109/ICDCSW.2011.20

Savitzky A, Golay MJE (1964) Smoothing and differentiation of data by simplified least squares procedures. Anal Chem 36(8):1627–1639. https://doi.org/10.1021/ac60214a047

Tanaka Y, Iwamoto K, Uehara K (2005) Discovery of time-series motif from multidimensional data based on MDL principle. Machine Learning - ML 58:269–300. https://doi.org/10.1007/s10994-005-5829-2

Torkamani S, Lohweg V (2017) Survey on time series motif discovery: time series motif discovery. Wiley Interdiscipl Rev: Data Mining Knowl Discov 7(2):e1199. https://doi.org/10.1002/widm.1199

Yeh CM, Zhu Y, Ulanova L, et al (2016) Matrix profile I: all pairs similarity joins for time series: A unifying view that includes motifs, discords and shapelets. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp 1317–1322, https://doi.org/10.1109/ICDM.2016.0179

Yoshimura G, Kanemura A, Asoh H (2019) Enumerating hub motifs in time series based on the matrix profile. In: Proceedings of 5th Workshop on Mining and Learning from Time Series (MILETS'19)

Zhu Y, Zimmerman Z, Senobari NS, et al (2016) Matrix profile II: exploiting a novel algorithm and GPUs to break the one hundred million barrier for time series motifs and joins. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp 739–748, https://doi.org/10.1109/ICDM.2016.0085

Zhu Y, Yeh CM, Zimmerman Z, et al (2018) Matrix profile XI: Scrimp++: time series motif discovery at interactive speeds. In: 2018 IEEE International Conference on Data Mining (ICDM), pp 837–846, https://doi.org/10.1109/ICDM.2018.00099

Zimmerman Z, Kamgar K, Senobari NS, et al (2019) Matrix profile XIV: Scaling time series motif discovery with GPUs to break a quintillion pairwise comparisons a day and beyond. In: Proceedings of the ACM Symposium on Cloud Computing. Association for Computing Machinery, New York, NY, USA, SoCC '19, p 74-86,https://doi.org/10.1145/3357223.3362721