Mining Recurring Patterns in Real-Valued Time Series using the Radius Profile

1st Dieter De Paepe
IDLab
Ghent University - imec
Ghent, Belgium
dieter.depaepe@ugent.be

2nd Sofie Van Hoecke
IDLab
Ghent University - imec
Ghent, Belgium
sofie.vanhoecke@ugent.be

***Abstract*—Time series analysis is becoming more popular in both research and industry. One recent innovation is the Ostinato algorithm, which finds the best preserved patterns that are repeated in a collection of series, i.e. consensus motifs and corresponding radii. However, Ostinato only works as a batch algorithm, can only find the top-k patterns, only finds patterns that are repeated in multiple series and has a runtime that depends on the input series and setup parameters. To tackle these limitations, we present two algorithms in this paper that can answer broader questions. First, we created an anytime version of Ostinato, called Anytime Ostinato, that finds the exact consensus radius for each subsequence, i.e. the radius profile, or can estimate these radii in a fraction of the time. Second, we designed a batch algorithm, called Single Series Ostinato, that finds the radius profile for a single series allowing us to detect repeating patterns in a single series, which is not possible for Ostinato. In this paper we explain both algorithms and apply them to the REFIT and PAMAP2 datasets respectively.**

***Index Terms*—time series analysis, motif discovery, consensus motif, common motif, matrix profile, radius profile**

I. Introduction

In the relatively new time series analysis domain, several topics are being researched today. These include broad topics like classification or anomaly detection, but also specialized techniques such as discord or motif discovery, that can act as building blocks for other techniques.

Whereas many publications deal with finding the best matching subsequence (motif discovery) in time series, only one (recent) publication tackles finding the most similar subsequence in a collection of series, the so called *consensus motif*. Specifically, given a collection of series and a subsequence length, the Ostinato [1] algorithm finds the subsequence s and corresponding minimal distance d (the *radius*), so that for each series the distance from s to the best matching subsequence of that series is d or less. A straightforward variant of Ostinato allows us to find the top-k consensus motifs instead of only the top-1.

This research was partly funded by the imec.icon project PROTEGO, which is co-financed by imec and VLAIO.

However, the Ostinato algorithm has two important restrictions. First of all, it assumes a good match is to be found in every series. If this is not the case, the algorithm will take a substantial longer time and may produce unintuitive results. A variant that drops this assumption exists but requires insight in the data and a longer runtime. Second, Ostinato assumes we only care about the single best match in every series. This means that multiple occurrences of a pattern within a single series are disregarded. As a result, we cannot use Ostinato to find the most common pattern in a (collection of) series where we do not know in advance where the pattern occurs.

To solve these shortcomings we present two innovations. First, we present an anytime version of the Ostinato algorithm that finds the consensus radius for *all* subsequences, i.e. the *radius profile*, rather than the top-k. Due to the anytime property, we can choose to spend a longer runtime for more accurate results. Still, we show that we get representative results for all subsequences in less time than it takes Ostinato to find the top-1 result and that in some cases the complete, exact calculation is faster than the top-1 Ostinato calculation. Secondly, we present an algorithm that finds the radius profile for a single series, allowing us to find repetitions in a series or in multiple series while ignoring where these repetitions occur.

The source code for our algorithms and the experiments listed in this paper have been made available online [2], as well as an extended version of this paper containing additional visualisations and pseudo code.

II. Related Work

To the best of our knowledge, little to no work can be found regarding repeating patterns in real-valued series. Motif discovery techniques, such as Matrix Profile [3], aim to find the best matching subsequence pair rather than the most common one. However, the best match is not guaranteed to also be part of the most common pattern-group. An adaptation of the Matrix Profile technique has been used to find time series chains, which are repeating patterns that slowly change throughout time [4]. Again, there is no guarantee that these chains will overlap with the best repeating pattern, because time series chains

2374-8486/20/$31.00 ©2020 IEEE
DOI 10.1109/ICDM50108.2020.00113

Authorized licensed use limited to: Politecnico di Torino. Downloaded on March 31,2021 at 17:26:58 UTC from IEEE Xplore. Restrictions apply.

assume a gradual changing pattern. Another variation is the Time Series Snippets algorithm [5], which looks for representative subsequences to summarize a series. However, this algorithm only considers a subset of all subsequences and will evaluate similarity against the entire dataset, making it unsuited for cases where we want to find well-preserved patterns with few repetitions. From the music domain, REPET is a Fourier based technique to separate foreground audio from repeating background music [6]. The technique assumes the pattern is repeated continuously, as is often the case in background music, making it unsuited to find patterns that are spread throughout a series. To conclude, it seems only the Ostinato algorithm, described in the recent work by Kamgar et al. deals with finding repeated patterns in multiple real-valued series [1].

III. The Ostinato Algorithm

In this section, we provide an overview on how Ostinato works and highlight some of its properties. However, we first introduce the distance matrix as a visual aid for this and later sections.

Given a number of time series $S_1, \ldots, S_n$ and a subsequence length m, we can represent the pairwise distance of all subsequences as a distance matrix D, as shown in Figure 1. Here, both axes represent all possible subsequences. That is, element $D[i, j]$ contains the (z-normalized Euclidean [3]) distance between the i-th and j-th subsequence.

Figure 1 also shows the relation between the distance matrix and a consensus motif. For any subsequence c, we can calculate the distance between c and the best matching subsequence of all series (i.e. d_1, d_2, d_3 and d_4 in Figure 1). The maximum value of these distances is defined as the radius r. The top-k consensus motifs are the k subsequences for which the radius is minimal.

Ostinato finds the top-1 consensus motif and corresponding radius using a straightforward greedy branch and bound approach. For series S_1, it calculates all subsequence distances between S_1 and S_2, tracking the best distance for each subsequence of S_1. Next, the subsequence with the lowest distance is considered the candidate consensus motif and its radius is calculated by determining the distance with all other series. At this point, Ostinato has an upper bound and repeats the process for all other subsequences of S_1, aborting the search if at any time the upper bound is passed. After all subsequences of S_1 have been considered, the search is repeated for the next series. Pseudocode and a visual representation of this process can be found in the extended version of this paper [2].

Due to the branch and bound approach, the runtime of Ostinato is dependent on details of the time series. In a very bad case, Ostinato has to calculate a large portion of the distance matrix (excluding the self-join of each series). In the very best case, it has to calculate n full-series calculations and *one* single column. In more realistic cases, the number of columns calculated will depend on

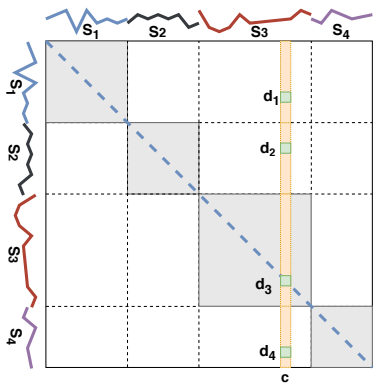

Fig. 1. Distance matrix for four series $S_1, \ldots, S_4$. Each row i and column j represents a single subsequence of one series and the element at index $[i, j]$ corresponds to the distance between those subsequences. As a side effect of this, the main diagonal (dashed blue) consists of all zeros. The consensus radius of a specific subsequence c is calculated by finding the best matching subsequence for all series and taking the maximum of these distances $(d_1, \ldots, d_4)$. The top-k consensus motifs are the k subsequences with the minimum radius.

the relation between the found upper bound and any distances that remain to be calculated. As an unfortunate side effect of this, the performance of Ostinato degrades when a specific pattern is repeated in all series but one, for example due to a labeling error. Not only will this cause a poor result, but the calculation will also take longer to finish due to the higher upper bound.

We demonstrate this is in Figure 2, where we compare the timings of four different collections of time series. The first collection consists of 10 identical series, and represents the optimal case for Ostinato. The second collection consists of 10 series of random noise, where we expect a large upper bound due to the lack of patterns. The third collection represents real-world data and was extracted from the REFIT power consumption dataset (aggregate readings from the first house) [7], with each series corresponding to a different time range. As shown later on, this dataset contains multiple repeating patterns. The final collection was the same as the third, but with one series replaced by randomly sampled noise, as to mimic an incorrectly labeled time series. The length of each series was chosen as a multiple of two, and the subsequence length was 1024, as was used in the benchmarks of the original paper [1].

As expected, the timings in Figure 2 show a similar trend, but vary greatly between the different types of time series. It is somewhat surprising how the longest runtime does not belong to the random dataset, but rather to the realistic dataset where one series was replaced with noise. In this case, the runtime is about 16 times longer than the optimal case. We suspect this is due to the large amount of good matches, all of whose distances will be obtained through column-calculations, only to end up with a poor match in the random series, whereas the pure random data

Authorized licensed use limited to: Politecnico di Torino. Downloaded on March 31,2021 at 17:26:58 UTC from IEEE Xplore. Restrictions apply.

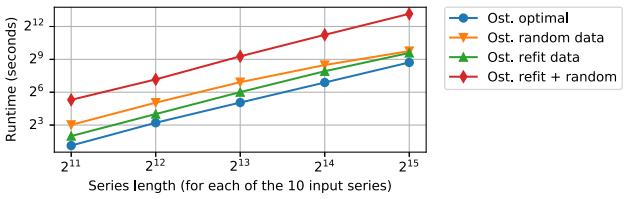

Fig. 2. Timings for Ostinato on four different types of time series data for subsequence length 1024. Each time, the input consisted of 10 series of the same length. We see that the type of data has a major influence on the runtime, due to the branch-and-bound approach.

will have few good matches overall.

Each of the n full-joins are calculated in $O(l^2)$ using the STOMP [8] algorithm, and each column is calculated using the MASS algorithm in $O(l \log l)$ [9], with n representing the number of series and l the average series length. Overall, we can say the memory complexity is $O(l)$ and the time complexity is $O(n^2 l^2 \log l)$. Note that the subsequence length m does not affect the runtime.

Two variants of Ostinato are mentioned in the original paper [1]. The first variant lets Ostinato find the top-k consensus motifs by tracking the k best results instead of the single best one. Overall, this will result in a higher upper bound and more column-wise calculations in the distance matrix. The second variant can be used to find the top-1 k of n consensus motif, namely the best consensus motif that can be found in a subset of k series. To do this, a full join is calculated and best matches are tracked for each series to $n - k + 1$ other series, rather than one.

As we have shown, Ostinato is an exact and fast algorithm for finding the top-1 (or top-k) consensus motifs. However, there are still some limitations. While it could return intermediate solutions, Ostinato remains a batch algorithm [1], which might be unsuited for applications with strict time constraints. Secondly, the upper bound calculation differs for the top-k and k of n variants, meaning that the user needs to have an idea of what he is looking for before starting a calculation. In the next section, we present an anytime version of Ostinato to calculate not only the top-k motifs, but *all* consensus motifs, including *all* k of n variants, in a single run.

IV. Finding All Consensus Motif Radii

Finding all consensus motifs rather than the top-k might give further insights into the properties of the series. This is similar as to how the Matrix Profile [3] was originally introduced as a way to find all motifs rather than only the top motif, but has since seen numerous other applications including series segmentation, visualization or rule mining. In a way, it does not make sense to talk about finding all *consensus motifs*, since every subsequence is one. Instead, in this section we will discuss how to find the corresponding radius for each subsequence, i.e. the *radius profile*.

Looking back at Figure 1, we could simply calculate the full distance matrix (excluding the diagonal blocks),

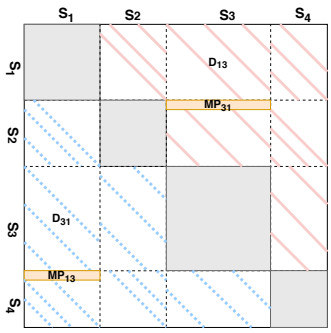

Fig. 3. Visual representation of the distance calculation for the Anytime Ostinato algorithm. Diagonals are calculated for each pair of series in the upper triangle of the distance matrix (red solid lines), and the distances are reused in the lower triangle (blue dotted lines). Whenever a diagonal is calculated for a pair of series (e.g. D_{13}), the two corresponding matrix profiles are updated (e.g. MP_{13} and MP_{31}). Note that we do not visualize all tracked matrix profiles.

and calculate the radius for each subsequence one after the other. This approach works and closely resembles the STAMP variant of Ostinato used by Kamgar et al. [1] for benchmarking, but is considerably slower than Ostinato. However, there is a better way to approach this problem.

A. Algorithm

Our solution consists of three parts. First, we can exploit the fact that the distance matrix is symmetric, i.e. the distance at index $[i, j]$ is the same as the distance at index $[j, i]$. This allows us to avoid half of all distance calculations. Secondly, for each subsequence, we track the distance to the best matching subsequence of the other series, i.e. the Matrix Profile for each pair of series. Finally, instead of calculating the distances column-wise (using STAMP or STOMP), we calculate diagonals using the SCRIMP [10] algorithm. A visual representation of our approach is shown in Figure 3.

By calculating diagonals, we effectively obtain an anytime calculation for the consensus radius of each subsequence. And, as we will demonstrate below, the anytime estimate on a small subset of the data still gives a representative radius estimate for each subsequence, from which we can also distill the top-k motifs.

Storing the matrix profile for each pair of series (excluding self-joins) provides all information to know the radius of each subsequence for both the top-k consensus motifs as well as the k of n variant. Furthermore, it allows us to resume refining the matrix profiles after looking at intermediate results, which could prove useful for data exploration purposes. One downside of this approach is the $O(n^2 l)$ memory usage, though this can be reduced to $O(nl)$ if the resumability property is dropped and we do not need the k of n variant, by processing the distance matrix in series-based batches.

Authorized licensed use limited to: Politecnico di Torino. Downloaded on March 31,2021 at 17:26:58 UTC from IEEE Xplore. Restrictions apply.

As our algorithm is stateful, it consists of an initialization, a calculation, and a result extraction step.

In the initialization step we iterate over all pairs of series from the upper triangle of the distance matrix, create a stateful class to calculate both matrix profiles and finally store the calculator and output variables. At this point, all matrix profiles contain only infinite distances.

The calculation step is straightforward: for each of the SCRIMP calculators (for which we refer to the SCRIMP paper [10]), we calculate diagonals until the desired fraction of all distances has been processed. For each diagonal calculated, the distances are used to update both corresponding matrix profiles.

At any point we can get the all-subsequence consensus radius by collecting the maximum distance per subsequence or the all-subsequence k of n consensus radius by collecting the $k-1$ lowest distances.

Pseudocode can be found in our extended paper [2].

B. Results on the REFIT Dataset

To demonstrate our technique, we use the REFIT dataset [7], which was also used in the original Ostinato paper. We extracted seven time series of the aggregated power consumption of the first house, each a day long (about 12843 data points), from the first week of December 2014. Using Ostinato, we calculated the top-1 consensus motif using a subsequence length of 800 (one and a half hours), which took 94 seconds. Next, we calculated the radius for *all* subsequences using our Anytime Ostinato algorithm for 5, 10, 25, 50 and 100 percent of the data. This took respectively 20, 36, 86, 170 and 328 seconds.

The results are shown in Figure 4. At the top, the radius profile for a single day is shown. For visual clarity, we have normalized the distances to a range of zero to one [11] and only show results for the smallest (five percent) and complete calculation. Even though the complete calculation used twenty times the amount of distance calculations, we see that the results are very similar. At the bottom of Figure 4, we see the top-3 consensus motifs for both calculations. Again, we can see that all resulting consensus motifs are very similar. In fact, the top-1 motif found using five percent of all calculations is simply a shifted version of the true top-1 motif. This means that we were able to obtain a representative estimate of the radii for *all* subsequences in less than a fourth of the time it took to obtain the assured top-1 motif using Ostinato.

Similarly we see how the top-2 and top-3 consensus motif calculated on 5% of the data closely resemble the true top-2 and top-3 motifs. In all cases, a better radius was found by examining the remaining 95% the data. Note that none of the 5% motifs are not wrong in the way that they never overestimate the radius. Instead, as more data is processed the radii for those patterns could still decrease or other patterns could be found to change the top-k listing.

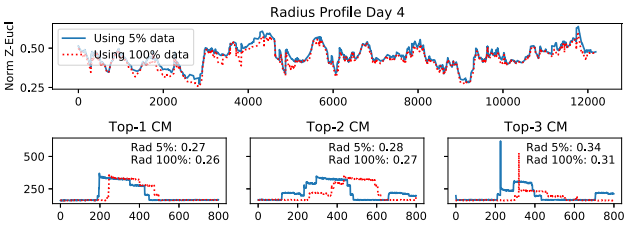

Fig. 4. Top: Radius profile for the fourth day (which contains the top-1 motif), using either five percent or using all distance calculations. We see that even when calculating only a small fraction, the profile matches very well with the exact profile. Bottom: the top-3 consensus motifs found in the fourth day of data using 5% (blue solid) or 100% (red dotted) of the calculations. Each motif corresponds to a local minimum in the radius profile.

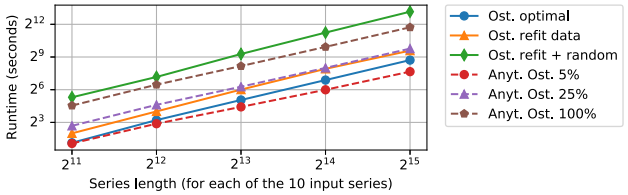

Fig. 5. Timings for Anytime Ostinato for various calculation percentages, overlayed with the Ostinato results from Figure 2. Note that the anytime algorithm is not affected by the type of data. Again, each run was done with 10 series of the same length and subsequence length of 1024. We see how the normal REFIT data takes about equally long as a 25% run, and how a complete 100% run is still twice as fast as the worst case runtime of Ostinato.

Next, let us look deeper at the runtime of our Anytime Ostinato technique. In Figure 5, we show the timings for various percentages of calculated distances, and overlay these with the Ostinato timings. Note that unlike Ostinato, the runtime for our algorithm is not affected by the type of data since we do not use a branch and bound method. As we can see, our 100% calculation is still faster than the worst result for Ostinato. For the most representative timing, we should look at the non-modified results for the REFIT dataset. In this case, we can calculate between 10 and 25 percent to get a similar runtime as Ostinato. However, our anytime algorithm returns a result for *all subsequences*, while Ostinato only finds the top-1 consensus motif.

V. Finding the Most Common Patterns

Both Ostinato and our improved anytime version are meant to find the patterns that are best conserved over multiple series, i.e. consensus motifs. Similarly, the Matrix Profile can be used to find the best preserved pattern within a single series, i.e. motifs. However, neither technique can be efficiently used to find the top-k best conserved patterns occurring multiple times in a single series. We could call these the intra-series consensus motifs, but to avoid confusion with the multi-series consensus motif, we opt to call these patterns *common motifs* instead.

Authorized licensed use limited to: Politecnico di Torino. Downloaded on March 31,2021 at 17:26:58 UTC from IEEE Xplore. Restrictions apply.

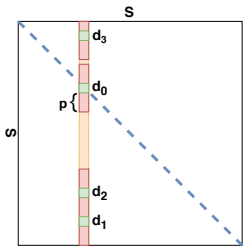

Fig. 6. Visualization of the link between the distance matrix and common motif radii. We calculate the distance for a single subsequence to the entire series (orange column). From this vector, we iteratively find the lowest distance (= best match), each time ignoring the p subsequences near any previous match (e.g.: $d_0, d_1, d_2, \ldots$). Since we ignore the orignal sequence (d_0), this means the common-1 radius equals d_1, the common-2 radius is d_2 and so on.

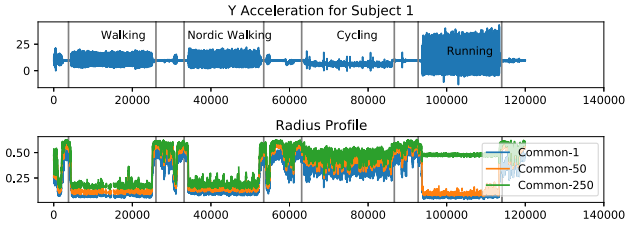

Fig. 7. Top: 20 minute extract of the PAMAP2 dataset consisting of four activities, separated by transition periods. Bottom: Single series radius profile, using a subsequence length of 100 (1 second). The radius differs depending on the activity being performed. The more repetitive the signal, the lower the radius. The radii are lowest for the walking and running activities, this means that the patterns are most similar through the data. The cycling activity has a high radius, meaning the subsequences are not repetitive.

At this point, there are two ways to formalize these common motifs. One way is to define a distance threshold and find the subsequences that have most matches under this threshold. The second way is to define a number of required matches and determine the subsequence that has the least distance from these matches. We opted for the second interpretation as it allows us to reuse the notion of a motif radius. In other words, for any subsequence s, we find the best k matches within the series and define the radius as the maximum distance of these matches. Then, the top-1 common-k motif is the subsequence for which this radius is minimal. To avoid the issue of trivial matches [3], where nearby sequences have similar distances, we require matches to be at least $p = \frac{m}{2}$ indices apart from each other and disregard the vicinity of the original subsequence s as well.

A. Single Series Radius Profile Algorithm

We show the relation between the distance matrix and the common motif radii in Figure 6. Again, the distance matrix is a square where each column or row represents a single subsequence from a series S. Note that we can also apply our technique to find the most common pattern in a set of series by simply appending the series, with a *nan* marker in between.

For a specific subsequence, we can calculate the distances to each subsequence in S, as shown by the orange column in Figure 6. From this distance vector, we iteratively look for the minimum value (= the best match), each time ignoring the subsequences within p positions of any minimum found so far, e.g. $d_0, d_1, d_2, \ldots$ If we ignore the distance from our original subsequence d_0, the other values are the the common-1, common-2, ... radii for this subsequence.

Again, the actual algorithm is straightforward. We iterate over all subsequences and for each subsequence we calculate the distance to all subsequences using the STOMP algorithm [8]. The STOMP algorithm calculates columns in the distance matrix similar to STAMP, but is faster when calculating neighbouring columns. Next, we iterate over all distances from lowest to largest. We count the number of non-excluded matches, mark the exclusion zone and skip excluded distances. Finally, we record the needed radii and skip to the next subsequence once all radii for the current subsequence are found.

Pseudocode can be found in our extended paper [2].

B. Results on the PAMAP2 Dataset

We demonstrate our technique on the PAMAP2 physical activity dataset [12], which contains accelerometer recordings of participants performing various activities. For demonstration purposes, we limit ourselves to an extract of the y-accelerometer of the first subject performing four different activities, as shown in Figure 7 (top).

We calculated the common-k radius profile for values 1, 50 and 250 for subsequence length 100 (i.e. 1 second of recording). These profiles are shown in Figure 7 (bottom). Again, the distances have been normalized.

We see how the walking, nordic walking and running activity have low radii, meaning that the corresponding signals are repetitive. One exception here is the high common-250 radius for the running activity, meaning that there are less than 250 good matches to each subsequence, which is most likely a result of the repetition interval and the exclusion zone for trivial matches. While the radius profile shows how repetitive each part of the signal is, it gives no direct insight as to which parts are similar to each other. Fortunately, we can use the radius profile to mine for common patterns with some additional work.

C. Finding the Top-k Common Motifs

The top-1 common-k motif is easily determined: it is the subsequence for which the common-k radius profile is minimal. However, finding the top-k patterns is a little more challenging. Consider we have found the top-1 common-k pattern p_1 and also know its k matching subsequences $m_1, \ldots, m_k$. For the top-2 pattern, we obviously exclude p_1. As $m_1, \ldots, m_k$ are very similar to p_1, it makes sense to also exclude these as well. However, there might be other sequences than $m_1, \ldots, m_k$ that are similar to p_1,

Authorized licensed use limited to: Politecnico di Torino. Downloaded on March 31,2021 at 17:26:58 UTC from IEEE Xplore. Restrictions apply.

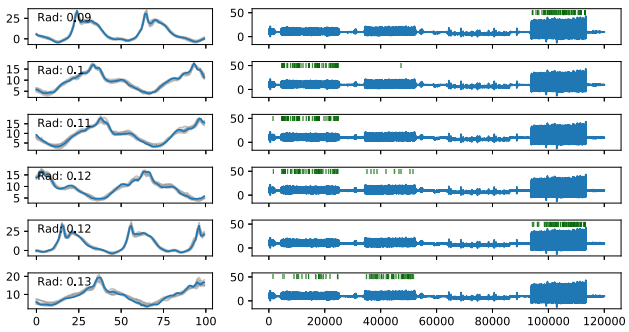

Fig. 8. The top-6 common-100 motifs for the PAMAP2 dataset. On the left we show the motif (blue) overlayed with the 10 best matches (light gray). On the right, we show the locations of the 100 matches that fall within the radius of the motif. We see that the common motifs originate from the three activities with repetitive behavior. Some motifs appear similar to each other, which can be explained due to slight variations of the activity speed, but also due to an imperfect similarity threshold.

perhaps having some of their k best matches overlap with $m_1, \ldots, m_k$. Ideally, we want to exclude *all* matches to p_1 that we consider as too similar, though we do not know in advance where exactly this threshold is.

Our approach for finding the top patterns is straightforward. We start by finding the top-1 pattern. Next, we calculate the distances between this pattern and all subsequences in the series using MASS. Whenever a subsequence is a good match (and thus is below a predefined similarity threshold), we exclude both the subsequence as well as the neighbouring subsequences, by changing the radius to infinity. This process is repeated until no more motifs can be found. Pseudocode can be found in the extended version of this paper [2].

D. Common Motifs in the PAMAP2 Dataset

Figure 8 shows the top-6 common-100 motifs for the PAMAP2 dataset, along with the location of each of the 100 matches in the data. We see that the top-1 and top-5 motif are very similar and correspond to the running activity. The other four motifs are shared between the walking and nordic walking activity. While the occurrences show how the patterns are distinguishable between these activities, there are overlaps, most noticeable for pattern six. We used a distance threshold of 0.14, which was determined manually after inspecting the first motif.

VI. Conclusion

In this paper we tackled the question on how to find repeating subsequences in one or more time series containing real values. For this, we have introduced a new time series primitive, i.e. the radius profile. This radius profile contains the radius for each subsequence in a series, where the radius is the maximum distance needed to reach a predefined number of best matches.

A first case concerns finding repetition across multiple series. For this we have extended the Ostinato algorithm, which calculates the top-1 radius, to an anytime version that calculates all radii. Our Anytime Ostinato algorithm can calculate an exact profile or make an estimate in a fraction of the time. Even an exact calculation can still be faster than Ostinato for some types of input data. Using the consensus radius profile, one can easily find the k patterns that are most similar over (a subset of) the series collection, i.e. the consensus motifs.

A second case concerns finding well-preserved repetitions in one or more series, irrespective of where they are repeated. Here, we have introduced a new algorithm to find the common-k radius profile, where k represents a predefined number of matches. The profile can be used to visualize the repetitive nature of data and to find the patterns that are repeated in a series, i.e. common motifs.

References

[1] K. Kamgar, S. Gharghabi, and E. Keogh, "Matrix profile XV: Exploiting time series consensus motifs to find structure in time series sets," *Proc. - IEEE Int. Conf. on Data Mining, ICDM*, vol. 2019-November, no. Icdm, pp. 1156–1161, 2019.

[2] [Online]. Available: https://sites.google.com/view/mining-patterns-radius-profile

[3] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, and E. Keogh, "Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets," in *2016 IEEE 16th Int. Conf. on Data Mining (ICDM)*. IEEE, dec 2016, pp. 1317–1322.

[4] Y. Zhu, M. Imamura, D. Nikovski, and E. Keogh, "Matrix Profile VII: Time Series Chains: A New Primitive for Time Series Data Mining," in *2017 IEEE Int. Conf. on Data Mining (ICDM)*. IEEE, nov 2017, pp. 695–704.

[5] S. Imani, F. Madrid, W. Ding, S. Crouter, and E. Keogh, "Matrix Profile XIII: Time Series Snippets: A New Primitive for Time Series Data Mining," in *2018 IEEE Int. Conf. on Big Knowl. (ICBK)*. IEEE, nov 2018, pp. 382–389.

[6] Z. Rafii and B. Pardo, "REpeating pattern extraction technique (REPET): A simple method for music/voice separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 21, no. 1, pp. 71–82, 2013.

[7] D. Murray, L. Stankovic, and V. Stankovic, "An electrical load measurements dataset of united kingdom households from a two-year longitudinal study," *Scientific data*, vol. 4, no. 1, pp. 1–12, 2017.

[8] Y. Zhu, Z. Zimmerman, N. S. Senobari, C.-C. M. Yeh, G. Funning, P. Brisk, and E. Keogh, "Matrix Profile II : Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins," *2016 {IEEE} 16th Int. Conf. on Data Mining (ICDM)*, pp. 739–748, 2016.

[9] A. Mueen, Y. Zhu, M. Yeh, K. Kamgar, K. Viswanathan, C. Gupta, and E. Keogh, "The fastest similarity search algorithm for time series subsequences under euclidean distance," August 2017, http://www.cs.unm.edu/ mueen/FastestSimilaritySearch.html.

[10] Y. Zhu, C.-C. M. Yeh, Z. Zimmerman, K. Kamgar, and E. Keogh, "Matrix Profile XI: SCRIMP++: Time Series Motif Discovery at Interactive Speeds," in *2018 IEEE Int. Conf. on Data Mining (ICDM)*. IEEE, nov 2018, pp. 837–846.

[11] D. De Paepe, D. Nieves Avendano, and S. Van Hoecke, "Implications of z-normalization in the matrix profile," in *Pat. recog. applications and methods, 8th Int. Conf., ICPRAM 2019, Revised Selected Papers*, M. De Marsico, G. Sanniti di Baja, and A. Fred, Eds., vol. 11996. Springer, 2020, pp. 95–118.

[12] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *Proceedings - International Symposium on Wearable Computers, ISWC*, 2012, pp. 108–109.

Authorized licensed use limited to: Politecnico di Torino. Downloaded on March 31,2021 at 17:26:58 UTC from IEEE Xplore. Restrictions apply.