Tiered Clustering for Time Series Data

Ruizhe Ma[1(✉)] and Rafal Angryk[2]

[1] University of Massachusetts Lowell, Lowell, MA 01854, USA
ruizhe_ma@uml.edu
[2] Georgia State University, Atlanta, GA 30302, USA

Abstract. Clustering is an essential unsupervised learning method. While the clustering of discrete data is a reasonably solved problem, sequential data clustering, namely time series data, is still an ongoing problem. Sequential data such as time series is widely used due to its abundance of detailed information. Often, normalization is applied to amplify the similarity of time series data. However, by applying normalization, measurement values, which is an important aspect of similarity, are removed, impairing the veracity of comparison. In this paper, we introduce a tiered clustering method by adding the value characteristic to the clustering of normalized time series. As such, two clustering methods are implemented. First, the Distance Density Clustering algorithm is applied to normalized time series data. After obtaining the first-tier results, we apply a traditional hierarchical clustering of a summarized time series value to further partition clusters.

Keywords: Unsupervised learning · Cluster · Time series

1 Introduction

The majority of data used in traditional data analysis are discrete point data, either an instantaneous point value (i.e., point in time) or a summarized point value (i.e., average). While point data is efficient to store and process, the obvious drawback is the lack of rich details. On the other hand, sequential data contains much more details on the process of a recorded event. Time series is a special type of sequential data, it is ordered and evenly spaced sequential values. Time series is extensively applied in various real-world applications.

Clustering is an important part of exploratory data mining; essentially, it is the partitioning of data to have high within-cluster similarity and low between-cluster similarity. A clustering process can be an independent procedure to gain insight into the distribution of a dataset or as a pre-process or subroutine for other data mining tasks, such as rule discovery, indexing, summarization, anomaly detection, and classification [6]. The application of clustering is very diverse; it can be applied in fields such as pattern recognition, machine learning, bioinformatics, and more.

Cluster analysis is a reasonably well-studied problem in the data mining community. The clustering of time series, however, is a relatively newer facet. Due

I. Awan et al. (Eds.): Deep-BDB 2020, LNNS 309, pp. 1–12, 2022.
https://doi.org/10.1007/978-3-030-84337-3_1

to the high dimensionality of time series data, the distribution can be difficult to comprehend. The predetermination of cluster parameter settings is already a difficult task with discrete data. The global parameter used for partitioning can be even more complicated to identify for time series data, as it is near impossible to visualize the respective position and correlation of time series. Therefore cluster algorithms with minimal parameter settings are more beneficial, making the study of the effect of different hierarchical structures of a time series dataset an important aspect to consider.

Previous work on solar flares has shown that profiles could be used to identify the otherwise unnoticeable distinction amongst time series data [12]. A key application of time series profiles is prediction. If distinct trend profiles can be identified prior to the occurrence of an event, then predictions can be made as new measurements come in, near realtime. Another possible application of time series cluster profiles is identifying possible sub-classes within existing identified classes. If different profiles within an existing class can be found, this would insinuate the existence of physical sub-classes within the current definitions. Both applications may be hard to achieve using discrete point values, whereas the adoption of time series data and shape-based analysis could set the stage in this direction.

The issue with using normalized time series data to generate cluster profiles is that in the process of normalization, certain aspects of time series characteristics are lost. There are three aspects of time series similarity: range value similarity, duration similarity, and shape similarity [12]. By normalization, the shape similarity is amplified while sacrificing value similarity. Therefore, even though the shape similarity is much more apparent and that clustering algorithms can build clusters of more similar time series, accuracy did not improve when compared to clustering with data that is not normalized. In this paper, we extend the normalized cluster profiles by adding another layer of value-based clustering, in the hope of combining two types of similarity and generating better results.

The rest of this paper is organized as follows: Sect. 2 presents the related work. Section 3 discusses the applied clustering methods and how they produce tiered cluster results. Section 4 briefly discuss the solar pre-flare time series data used in our experiments. Section 5 presents the results and analysis. Finally, Sect. 6 summarizes this paper.

2 Background

2.1 Distance Measure

Real-world events are complex and detailed, often times when we evaluate events on summarized values we trade preciseness for efficiency. With the improved storage and processing capabilities, sequential data has gained more popularity. Time series is a popular type of sequential data, it is a sequence of measurements that are equally spaced in time. Since real-world events are complex and often affected by a multitude of unforeseeable external factors, it is highly probable to observe differences for both duration and measurements for time series describing

the same class of events. Therefore the similarity determination for time series is not a trivial problem.

There are two main types of similarity measure, lock-step and elastic. The traditional lock-step similarity measure L_p norm refers to the Minkowski distance raised to the power of p. Minkowski distance is most commonly used with L_p where $p = 1$ (Manhattan distance), and with L_p where $p = 2$ (Euclidean distance). Euclidean distance is the straight-line distance. When applied to time series, assuming we are working with equal length time series, Euclidean distance will always be made based on a one-to-one mapping where the i-th element in one sequence is always mapped to the i-th element in the compared sequence. Comparatively, elastic measures allow one-to-many as well as one-to-one mappings [9]. Originally used in the field of speech recognition, the Dynamic Time Warping (DTW) algorithm is one of the most widely used elastic similarity measurement [1–3,7]. DTW enables computers to find an optimal match between two given sequences under certain constraints, and it allows a flexibility in sequential similarity comparisons.

Euclidean and DTW distances [5] of given time series Q and C are shown in Eq. 1 and 2, respectively, where time series Q: $Q = \{q_1, q_2, ..., q_i, ..., q_n\}$, and time series C: $C = \{c_1, c_2, ..., c_j, ..., c_m\}$.

$$Dist(Euclidean) = \sqrt{\sum_{i=1}^{N}(q_i - c_i)^2} \tag{1}$$

$$Dist(DTW) = min\{W(Q, C)\} \tag{2}$$

When Euclidean distance is used for time series data, the total distance is the sum of distances between each of the one-to-one mapping between elements q_i and c_i. In the case of DTW, however, a $n \times m$ distance matrix is first constructed containing all possible distances for each q_i and c_i pairing. Then each optimum step is chosen to form the optimal path, among the numerous warping paths of $W = w_1, w_2, ..., w_k, ..., w_K$, the path that minimizes the mapping between time series Q and C, represented as $min\{W\}$, is considered as the optimal warping path.

At each step of the DTW algorithm, several choices are presented, and the allowed possibilities is referred to as the step pattern. The ability to choose a minimal step translates to data point mapping, and this choice gives the ability and effectiveness in finding shape similarities in time series data. Equation 3 is considered as one of the most basic and commonly used step patterns. Here the cumulative distance $D(Q_i, C_j)$ is the sum of the current distance $d(q_i, c_j)$ and the minimum distance from the adjacent elements.

$$D(Q_i, C_j) = d(q_i, c_j) + min \left\{ \begin{matrix} d(Q_i, C_{j-1}) \\ d(Q_{i-1}, C_{j-1}) \\ d(Q_{i-1}, C_j) \end{matrix} \right\} \tag{3}$$

For both clustering and cluster representation, an effective time series averaging technique is required. Here we use the time series averaging technique

DTW Barycenter Averaging (DBA) [8]. Instead of dividing the summation, as is with traditional averaging, DBA considers shape by using DTW to minimize the Within Group Sum of Squares (WGSS). Simply put, given a time series set of $\mathbb{S} = \{S_1, S_2, ..., S_n\}$, the time series $C = \{c_1, c_2, ..., c_t\}$ is considered an average of $\mathbb{S}$ if it minimizes:

$$WGSS(C) = \sum_{k=1}^{n} dtw(C, S_n)^2 \tag{4}$$

2.2 Time Series Similarity

In different applications, the similarity of time series can vary. The three key elements of time series similarity are *range value similarity*, *duration similarity*, and *shape similarity* [12]. Shown in Fig. 1, the range value similarity is demonstrated by sub-figure (a) and (c), it refers to the absolute range value of time series, this similarity signifies the vertical comparability of two given time series. The duration similarity is demonstrated by sub-figure (a) and (b), it refers to the time series measurement duration, this similarity reveals the horizontal comparability of two given time series. Demonstrated by sub-figure (b) and (c), the shape similarity focuses more on the contour of the given time series.

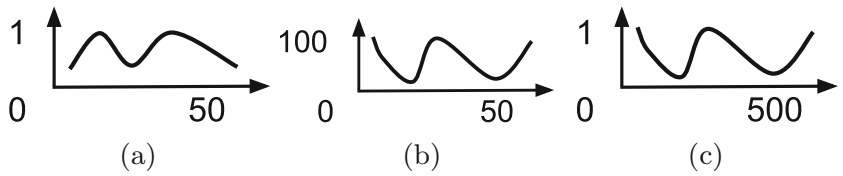


Fig. 1. Time series similarity: (a) and (c) demonstrate range value similarity; (a) and (b) demonstrate duration similarity; (b) and (c) demonstrate shape similarity.

The similarity of time series is highly contextual and has broad applicability. Therefore, certain aspects of similarity could be deemed more significant under certain circumstances. However, all three elements of similarity should be fulfilled for two time series to be considered truly similar. Therefore, for unsupervised learning, we have to consider which similarity feature is applied to for clusters.

2.3 Normalization Methods

Normalization is often used to scale data so that the data will fall within a specified range. In addition, time series normalization can also be used to shift and scale data to eliminate the effect of gross value influences. Evidently, normalization is not suitable for all time series data; it is more useful when the values are on different range or when the value differences are substantial enough for certain details to be overlooked. The four most commonly applied normalization

techniques for time series data are *Offset Translation*, *Amplitude Scaling*, *Trend Removal*, and *Smoothing*. When a certain normalization is applied, the same normalization is applied to all the time series in the dataset.

Offset Translation. Offset translation is the vertical shift of time series; it was originally used in signal processing when sequences are similar in shape but are within different ranges.

$$ts = ts - mean(ts) \tag{5}$$

Here the mean value is independently computed for each time series and is the average over all the values in that specific time series. The translation of the offset can be useful for similarity comparisons. However, an immediate drawback of this operation is that the range values would be eliminated since the value differences are removed. This, however, can be made up for in the second stage of our tiered clustering process.

Amplitude Scaling. Amplitude is another term from signal processing; it measures how far and in which direction a variable differs from a defined baseline. Scaling of a signal's amplitude means changing the strength of the signal. With time series data, we remove the different amplitudes in hopes of finding similarity by excluding the strength of the physical parameters.

$$ts = (ts - mean(ts))/std(ts) \tag{6}$$

Shown in Eq. 6, amplitude scaling is achieved by first moving time series by its mean and then normalized by the standard deviation. Which means that offset translation is included in amplitude scaling. In fact, when $std(ts) = 1$, the two methods are identical.

Trend Removal. Trend removal is mostly applied in prediction models. Trends represent long-term movements in sequences. Trends can be distracting when attempting to identify patterns in sequential data, and therefore, it is often justified to remove them for revealing possible oscillations. To this end, the regression line of the time series needs to be identified and then subtracted from the time series. Unlike offset translation and amplitude scaling, trend removal is not a straightforward operation. In practice, there could be various types of trends or even multiple trends. In our experiments, we only considered the simple linear trend and the logarithmic trend.

Smoothing. Smoothing is performed with a moving window on the time series to obtain the average values of each data point with those of its neighbors. While it can eliminate some irregular movements, it can be sensitive to outliers and also invalidates data at the beginning and the end of any time series.

In the solar flare dataset for our experiments, the time series are relatively short in length (i.e., 60 data points) and is also noisy in nature. For a smoothing

window to be effective, the size is often relatively large. Therefore, an effective smoothing would excessively shorten the time series we are working with, rendering the result ineffective. For this reason, smoothing is not included in our experiments.

3 Tiered Clustering

Time series data is very domain-specific, meaning the data from one area could be processed in an entirely different way as the data from another field. Therefore, we use a tiered cluster method to encompass more dimensions of similarity. In this section, we present the cluster algorithms applied in our tiered clustering method, namely Distance Density Clustering (DDC) and Hierarchical Agglomerative Clustering (HAC).

3.1 Distance Density Clustering

The Distance Density Clustering (DDC) method [10] was specifically developed for time series clustering, and has shown promising results. Here we use it to cluster normalized time series. DDC is divisive in structure, meaning that performance generally increases as more clusters are introduced. In the extreme case of each event forming its own cluster, the method degenerates to a k-Nearest Neighbors algorithm with $k = 1$ (i.e., 1NN), where each instance of testing data is compared to all the existing training data, and assigned the label of its single closest neighbor. While setting k to 1 can drastically improve the classification accuracy, conceptually, 1NN is a memorization process and not a generalization process. Memorization processes are inherently less powerful in real-world applications, as a comparison against the entire historical archive is unrealistic in most circumstances.

While many existing clustering algorithms can be applied to time series, either with data summarization or effective distance measures, the effect is often limited. DDC typically generates more intuitive results for time series clustering [10], the main steps of which are shown in Algorithm 1. Initially, through majority voting, the furthest time series is identified and is used as the initial cluster seed. The furthest time series is the time series that is the furthest from the most number of other time series. Then the distances between all instances and the cluster seed are computed and sorted. The most significant increase in the sorted distances is considered as a virtual sparse region and is used to divide the dataset. Then new cluster seeds are identified, and the cluster assignment is re-balanced based on time series similarity. This process is iterated until no more clusters can be found, or the process has reached a user-defined threshold, such as a certain number of clusters have been generated. Finally, all the identified cluster seeds and their respective cluster elements are obtained.

Algorithm 1. Distance Density Clustering Algorithm

Require:

$E = \{e_1, ..., e_n\}$ is the time series events to be clustered
$C_{k-1} = \{c_1, ..., c_{k-1}\}$ is the set of cluster seeds
k is number of seeds
L_k is the cluster set of events based on the number of groups

$L_{k-1} \leftarrow Cluster(C_{k-1})$
$ar[1, 2, ..., k-1] = DistSort(L_{k-1})$
$value[i] \leftarrow max(ar[2] - ar[1], ..., ar[k-1] - ar[k-2])$
if $ar[n] - ar[n-1] == max(value[i])$ **then**
    $location[i] = n$
end if
if **then**$i \leftarrow max(value[1, ..., k-1])$
    $l(i_1, i_2) \leftarrow l(i), (c_{i_1}, c_{i_2}) \leftarrow c_i$
end if
return $L_n = \{1, 2, ..., i_1, i_2, ..., n\} \leftarrow C_k\{(c_1, c_2, ..., c_{i_1}, c_{i_2}, ..., c_n)\}$
for $e_i \in E$ **do**
    $(c'_1, c'_2, ..., c'_k) \leftarrow DBA(c_1, c_2, ..., c_{i_1}, c_{i_2}, ..., c_{k-1})$
    $UpdateClusterDBA(C_k)$
end for
return $C'_k = \{c'_1, ..., c'_k\}$ **as set of cluster seeds**
return $L_n = \{l(e) \mid = 1, 2, ..., n\}$ **set of cluster labels of** E

3.2 Hierarchical Agglomerative Clustering

Hierarchical clustering algorithm separates data into different levels that have a top to bottom ordering, which forms a corresponding tree structure. There are two types of hierarchical clustering, agglomerative, also known as Agglomerative Nesting (AGNES), and divisive, also known as Divisive Analysis (DIANA) [4]. AGNES is a bottom-up approach, where each event is assigned as its own cluster, and based on a specific linking mechanism, the most similar clusters are joint to form a new cluster. This process is repeated until all events are joint together. DIANA is a top-down approach, where all events start as a single cluster and are then partitioned to form two least similar clusters. This process is repeated until each event forms its own cluster. Both AGNES and DIANA are based on distance for measuring similarity.

In an agglomerative structure, clusters are joint based on the similarity between elements or clusters. When comparing the similarity of clusters, various measures can be adopted. The cluster merging method is called linkage. The most commonly used linkage measures use nearest, furthest, or average distance for cluster distance measurement, which corresponds to single link, complete link, and average link.

Both the DDC and the HAC are hierarchical in structure, but they form clusters based on different concepts. DDC takes advantage of a virtual sparsity split to form clusters, whereas HAC is purely based on distance/similarity split. Furthermore, HAC is a greedy approach and DDC is not. In our proposed

Algorithm 2. Hierarchical Agglomerative Clustering

Require:
set X of objectives $\{x_1, ..., x_n\}$
similarity function $dist(c_1, c_2)$
1: **for** $i = 1$ to n **do**
2: $c_i = \{x_i\}$
3: **end for**
$C = \{c_1, ..., c_n\}$
$l = n + 1$
4: **while** $C.size > 1$ **do**
$(c_{min1}, c_{min2}) = min\_dist(c_i, c_j)$ for all c_i, c_j in C
remove c_{min1} and c_{min2} from C
$C \leftarrow \{c_{min1}, c_{min2}\}$
$l = l + 1$
5: **end while**

clustering structure we take advantage of both cluster algorithms to focus on different aspects of similarity. We use DDC to focus on the shape similarity before using HAC to partition the data based on range similarity.

4 SWAN-SF Dataset

In this study, we use the Space Weather ANalytics for Solar Flares (SWAN-SF) [11], which is a benchmark dataset of multivariate time series (MVTS), spanning over a 9 year period (2010–2018). Essentially, the goal is to predict the most significant solar flare within the next 24 h with the 12 h of before-flare time series measurements for multiple parameters. For reference, 9 of the most interesting parameters are picked by domain experts and listed in Table 1. There are a total of 5 classes of solar flares, listed from quiet to the most powerful, FQ (flare-quiet), B class, C class, M class, and X class, and each time series is labeled with the most significant (largest) flare within the 24 h observation period. The most impactful flares are M and X class flares; therefore, in this paper, we are specifically focusing on the clustering of classes C, M, and X flares. This dataset can be considered as an MVTS dataset with 3 class labels, and the meaning of specific parameters should not interfere with the presented method.

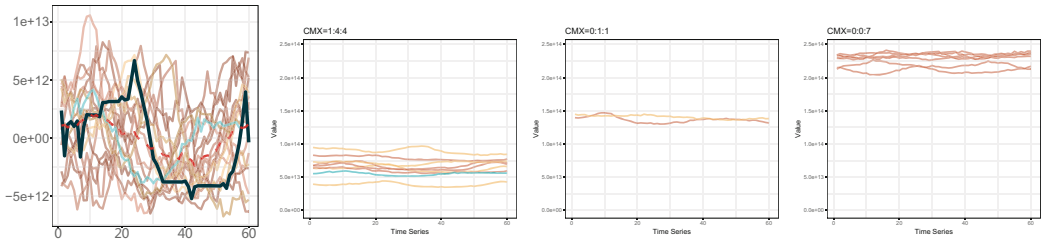
The measurements of solar flares cannot be clustered in a straight forward manner. While the duration of each event is the same, the range value similarity and the shape similarity can be challenging to be identified simultaneously. This is partly due to the vast variation of the strength of solar flare measurements. When the value of different events differs substantially, the shape details could become hard to distinct. In a previous study, shape-intuitive clusters were generated by clustering the normalized time series [12]; however, the accuracy performance was not improved despite the shape emphasis. This side effect of normalization can be eliminated by the actual value of different events. In this study, we are considering both the effect of shape similarity as well as the measurement values.

Table 1. Nine parameters selected by domain experts of which solar pre-flare time series are evaluated.

	Keyword	Description
1	MEANJZD	Mean vertical current density
2	MEANJZH	Mean current helicity
3	R_VALUE	Sum of flux near polarity inversion line
4	SAVNCPP	Sum of the modulus of the net current per polarity
5	SHRGT45	Fraction of area with shear angle >45°
6	TOTFZ	Sum of z-component of Lorentz force
7	TOTUSJH	Total unsigned current helicity
8	TOTUSJZ	Total unsigned vertical current
9	USFLUX	Total unsigned flux

5 Experimental Results

In consideration of fairness and to eliminate performance randomness, a balanced 5-fold cross-validation on the curated dataset of a total of 300 C, M, and X class instances was implemented. Cross-validation is a statistical evaluation method used to evaluate machine learning models where data is limited. The testing data is never included in the training process to avoid bias, and the training and testing are repeated for each data fold to ensure stability.



(a) Normalized cluster (b) Sub-cluster 1 (c) Sub-cluster 2 (d) Sub-cluster 3

Fig. 2. (a) shows one of the clusters generated by the DDC algorithm, (b), (c), and (d) are the sub-clusters generated by HAC.

First, we show in detail the advantage of applying a tiered clustering of normalized time series data with DDC and HAC. After processing normalized time series data with DDC, we apply HAC on each DDC generated cluster. Starting from the bottom of the dendrogram, when the branch ratio first exceeds the third quartile, we cut the dendrogram and obtain the corresponding clusters. The importance of both the shape and the measurement value is shown in Fig. 2.

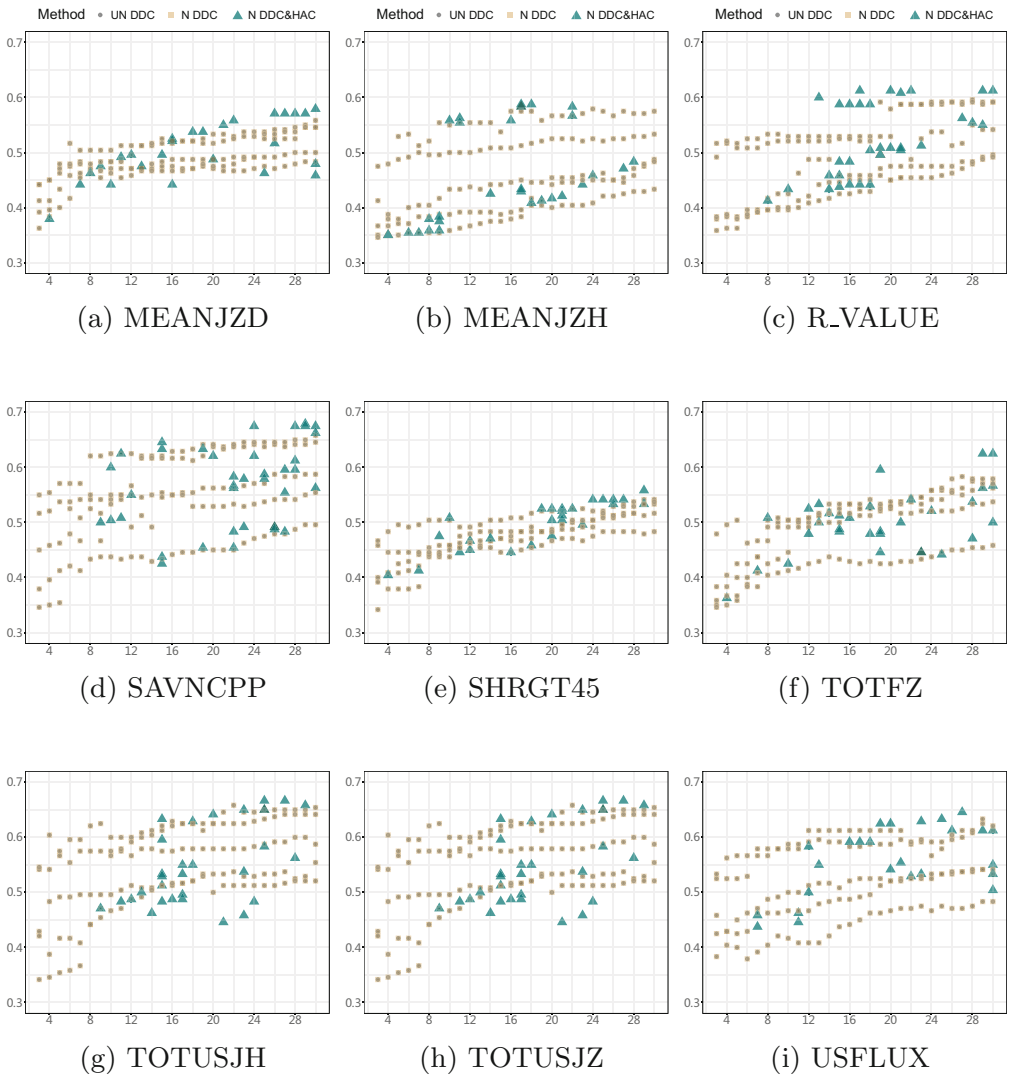


Fig. 3. Performance of different clustering approaches demonstrated by 9 parameters of solar pre-flare data. The x-axis is the increase of cluster numbers, and the y-axis is the corresponding accuracy value. Three cluster structures are shown, UN DDC is unnormalized DDC results, N DDC is normalized DDC results, and N DDC&HAC being normalized time series with DDC and HAC results.

A DDC generated DDC cluster is presented in Fig. 2(a), with the orange line being X-class flares, the yellow line being M-class flares, and blue being C-class flares, the time series average is the dark line. While this is not a pure cluster, the shape after normalization for all three classes is actually quite similar. Figure 2(b), (c), and (d) are the sub-clusters generated by HAC from the original cluster, here the actual value is taken into account. The ratio of classes C, M, and X is written above each sub-cluster. The third sub-cluster contains 7 X-class flares, the first and second are more assorted. However, considering both the shape similarity as well as the value similarity, it would be difficult even for

a human to distinguish the first and second sub-clusters just by the time series alone.

The overall performance of one fold is shown in Fig. 3, other folds are comparable in performance, but omitted for simplicity. Here the number of HAC are generically performed with dendrogram branch ratio, in practice HAC can be fine-tuned for different data or different parameters. For each parameter, the progression of accuracy improvement for each clustering method is demonstrated in relation to the number of clusters in Fig. 3(a)–(i). Different normalizations are all included in the figures. The unnormalized time series DDC results are referred to as "UN DDC", normalized DDC results are referred to as "N DDC", and normalized tiered clustering results from both DDC and HAC is referred to as "N DDC&HAC". The UN DDC accuracy results are overlapping with N DDC accuracy results. As concluded in the previous work [12], although the clustering of normalized time series generated more intuitive clusters, it did not improve the accuracy performance. This was partly due to the information loss in the normalization process. Therefore, when both the shape and the value information is considered, we see a general improvement in the tiered clustering structure with DDC and HAC, especially when the number of clusters increase.

6 Conclusion

Normalization is effective in finding shape similarities when the value differences are significant. However, in the process of normalization, measurement value information is lost. In this paper, we extend the clustering of normalized time series by reintroducing value information using hierarchical clustering. This way, we can take into account both the shape information as well as the value information embedded in the original time series measurements. We would like to note that this tiered clustering is not suited to all time series data, but an alternative method for time series data that may have extreme range value differences. This method could also be helpful in identifying new sub-classes within established data class in the future.

References

1. Sakoe, H.: Dynamic-programming approach to continuous speech recognition. In: 1971 Proceedings of the International Congress of Acoustics, Budapest (1971)
2. Myers, C., Rabiner, L.: A level building dynamic time warping algorithm for connected word recognition. IEEE Trans. Acoust. Speech Signal Process. **29**(2), 284–297 (1981)
3. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowl. Inf. Syst. **7**(3), 358–386 (2005)
4. Rokach, L., Maimon, O.: Clustering methods. In: Maimon, O., Rokach, L. (eds.) Data Mining and Knowledge Discovery Handbook, pp. 321–352. Springer, Boston (2005). https://doi.org/10.1007/0-387-25465-X_15

5. Müller, M.: Dynamic time warping. In: Müller, M. (ed.) Information Retrieval for Music and Motion, pp. 69–84. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74048-3_4
6. Chiş, M., Banerjee, S., Hassanien, A.E.: Clustering time series data: an evolutionary approach. In: Abraham A., Hassanien AE., de Leon F. de Carvalho A.P., Snášel V. (eds.) Foundations of Computational, IntelligenceVolume 6, vol. 206, pp. 193–207. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01091-0_9
7. Jeong, Y.-S., Jeong, M.K., Omitaomu, O.A.: Weighted dynamic time warping for time series classification. Pattern Recogn. **44**(9), 2231–2240 (2011)
8. Petitjean, F., Ketterlin, A., Gançarski, P.: A global averaging method for dynamic time warping, with applications to clustering. Pattern Recogn. **44**(3), 678–693 (2011)
9. Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., Keogh, E.: Experimental comparison of representation methods and distance measures for time series data. Data Min. Knowl. Disc. **26**(2), 275–309 (2013)
10. Ma, R., Angryk, R.: Distance and density clustering for time series data. In: 2017 IEEE International Conference on Data Mining Workshops (ICDMW), pp. 25–32. IEEE (2017)
11. Aydin, B., et al.: Multivariate time series dataset for space weather data analytics (manuscript submitted for publication). Sci. Data (2019)
12. Ma, R., Ahmadzadeh, A., Boubrahimi, S.F., Georgoulis, M.K., Angryk, R.: Solar pre-flare classification with time series profiling. In: 2019 IEEE International Conference on Big Data (Big Data). IEEE (2019)