2017 IEEE International Conference on Data Mining Workshops

Distance and Density Clustering for Time Series Data

Ruizhe Ma and Rafal A. Angryk
Georgia State University
Email: rma1@student.gsu.edu, rangryk@cs.gsu.edu

***Abstract*—Clustering is an important branch in the field of data mining as well as statistical analysis and is widely used in exploratory analysis. Many algorithms exist for clustering in the Euclidean space. However, time series clustering introduces new problems, such as inadequate distance measure, inaccurate cluster center description, lack of efficient and accurate clustering techniques. When dealing with time series data, Dynamic Time Warping (DTW) is an accepted and effective distance measure. For cluster updates and representation, DTW Barycenter Averaging (DBA) algorithm being a global averaging method using DTW and has proven to be an effective averaging method for time series data. In this paper, we propose a Distance Density clustering method that is a medoid-based clustering with time series data density consideration which provides clustering results in a hierarchy fashion. First, we introduce two clustering initialization techniques, from the time series similarity matrix we use majority voting to determine either the nearest or the furthest time series as the initial clustering seed. By doing so, our clustering method is deterministic, and the clustering results can always be reproduced. In the Distance Density clustering algorithm, we use medoids because it is a more representative alternative to the statistical mean, especially with time series data where the mean value is often non-existent. The time series density is a virtual density based on time series similarity; this can find more natural splits in a dataset and also the number of clusters does not need to be determined a priori. Experiments using the Distance Density clustering technique on the UCR dataset demonstrates that clustering initialization is crucial in obtaining stable and better results than random initialization on average, and is also more accurate than traditional distance clustering.**

***Index Terms*—Time Series Clustering, Density Clustering, Dynamic Time Warping**

I. INTRODUCTION

Cluster analysis is an unsupervised classification of patterns and is a well-studied problem in the data mining community [1]. Essentially, it is the task of grouping objects together where intra-cluster distance is minimized, and inter-cluster distance is maximized. Clustering is important for exploratory data mining and is applied in many fields, such as pattern recognition, machine learning, bioinformatics. Many clustering algorithms have been proposed over the years, and the clustering of discrete datasets can be considered a fairly solved problem with many well-established methodologies. However, existing clustering approaches do not perform well when applied to time series data, especially in the multidimensional case [2]. We explore the possibility of applying existing clustering ideas to time series data, which could help us to eventually develop efficient clustering algorithms specifically used for high-dimensional time series data.

Distance measure is a crucial measurement for any clustering algorithm. Its effectiveness directly impacts clustering accuracy. However, many traditional distance measures like Minkowski distance become irrelevant when comparing time series data. To effectively measure time series similarity, more attention needs to be focused on the shape of the data, rather than singular values. Dynamic Time Warping (DTW) was originally used for speech recognition and later introduced to other sequential data for similarity measurement. DTW has long been accepted as a good and efficient similarity measure for time series data [6]–[9]. DTW is effective for time series data because it can compare the similarity between time series of different lengths, and it allows acceleration and deceleration of sequences to a certain extent, it is therefore ideal for both synthetic and real-world time series data. The relevance of DTW has been demonstrated in various publications across several domains [10]–[14], and therefore our experiments will not include Euclidean measures.

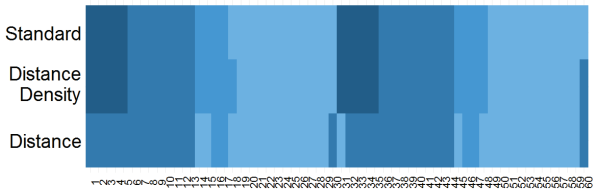

Figure 1. Comparison of standard clustering, Distance Density clustering, and k-means clustering using a heatmap. Time series are listed horizontally, and the color denotes which cluster the time series data belong.

Another obstacle for time series clustering is to find a representation of a cluster, often the average. However, calculating the average for time series data is not a trivial task [15], and the concept of non-time series data average is not the same as time series data. While standard clustering techniques can be applied to time series of uniform length, the shape factor of time series data is not accounted for [5]. If we apply the traditional averaging method on time series data, not only will it not work for different length sequences, but also the result will most likely not be representative of any of the original time series. Dynamic Time Warping Barycenter Averaging (DBA) was proposed by Petitjean et al. [15] as a way to calculate the average for time series data. By definition,

2375-9259/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDMW.2017.11

barycenter refers to the center of mass of two or more bodies orbiting each other. Here it refers to the application of using multiple time series sequences to calculate the average time series sequence. Since the averaging process uses DTW to map one sequence to another, the length of time series data is irrelevant and does not have to be uniform.

A common clustering algorithm is the k-means algorithm, because it is simple, easy to customize, and the results are intuitive and easy to interpret [3]. The goal of k-means clustering is to partition various observations into k clusters so that intra-cluster distances are minimized, and inter-cluster distances are maximized. A problem with traditional k-means is the use of the statistical mean as the cluster center. In any dataset, real-world or synthetic, it is very likely that the statistical mean could be non-existent, this is especially true for a time series dataset. Even when effective time series averaging techniques are applied, if a synthetic time series sequence never occurred, it is not a good representation for a group of time series data. Medoids on the other hand, are actual members of a dataset and are more commonly used as a more data-centric alternative to mean or centroid, notably for time series data [4], [5].

In this paper we will be discussing a distance density combined clustering algorithm and evaluate its performances. In addition to applying DTW and DBA with k-medoid, we propose two initialization techniques as well as a distance density clustering approach to avoid fluctuating clustering results as well as determining number of clusters k a priori. Fig.1 is a comparison of clustering results of standard class label, furthest seed initialized Distance Density clustering, and random initialized distance (k-means) clustering. The colors in the heatmap denotes the cluster the corresponding time series belongs to, we can see the clustering results from the furthest seed initialized Distance Density method is very close to the standard labels, and outperforming random initialized clustering. The main features of our work are outlined as follows.

Initialization: the very first and important step for a clustering algorithm is the initial seed selection. Since we can only achieve local optimum with k-medoid, the initialization step is crucial to the final clustering results. Here we pick the nearest and furthest point as the initial seed and compare it to a randomly selected seed.

Distance measure: various literatures have been dedicated to Euclidean distance and DTW comparisons, in this paper we will only give a short introduction and will not go into actual comparisons between the two distance measures and their impact on accuracy in the experiment section.

Cluster methodology: to avoid predetermining the k value and to find more natural splits in a dataset, we use the cluster seeds to obtain a distance graph, then by finding the point with the largest distance increase we are able to find the shift in cluster density, which could provide more intuitive clustering results.

Cluster update: we will be updating the clusters by finding the time series average using the DBA method, and finding the sequence that is the most similar to the average, here being the medoid.

The rest of this paper will be organized as follows, section II will introduce the background information as well as previous works; section III discuss our distance and density clustering technique; section IV contains the experimental results with various datasets; and finally section V concludes this paper and propose some future works.

II. Related Work and Background

A. Clustering Methodologies

When using DTW as the distance measure in clustering algorithms, the cost is the sum of DTW distance between each time series sequence and the cluster center. Theoretically, the optimal cluster center is found by simultaneously calculating the DTW path using all time series sequence within each cluster, which is a Steiner sequence problem. However, finding the Steiner sequence is proven to be NP-complete even in the discrete case [17], and multiple alignment is not tractable for more than a few sequences, and also requires unrealistic amount of memory [18].

K-means clustering was first introduced 50 years ago [1], and despite the fact that many other cluster algorithms have been proposed, it is still popular with many variations. It aims at minimizing the intra-cluster sum of squares, by using the proximity of objects to the medoids of the clusters formed by the algorithm [19]. Drawbacks of k-means include its poor performance in the presence of outliers, and unsatisfactory results when clustering non-globular data. Based on the original k-means, k-medoid clustering was introduced by Kaufman and Rousseeuw [20], where instead of using the mean value, which could very likely be an artificial data point, it uses existing data points as clustering centers. In the case of time series clusters, we find the sequence that has the shortest DTW distance to the time series average sequences. Compared to the standard k-means, k-medoid is more robust to noise and outliers. Using a medoid time series that actually occurs to represent the time series cluster is a more data-centric choice.

As k-means is an NP-hard problem [21], finding the exact solution for large datasets is near impossible. The common approach is using Lloyd's algorithm, which finds the approximate solution. The largest issue with this approach is that it achieves a local optimum, meaning the seed initialization heavily influences the end result. With randomly chosen initial seeds, the quality of clustering could fluctuate, therefore any meaningful help with initialization is better than random. Even though sometimes random initialization can result in higher accuracy, deterministic clustering can provide a stable algorithm with comparable results. This is especially important in applications, where stability and reproducibility is just as important as accuracy. To overcome the fluctuation of results due to random initialization, k-means++ was proposed by Arthur et al. [22], it is an approximation algorithm that can help to avoid some poor clustering for the original k-means. The idea behind the k-means++ algorithm is to initialize centroids (medoids) far away from each other, this speeds

up the algorithm and provide better cluster results [22]. In this study we propose and test furthest seed initialization and nearest seed initialization for time series data, it will be compared to the results with that of the random case.

Although simple to execute, neither k-means nor k-medoids is known for its high accuracy. Density-based spatial clustering of applications with noise (DBSCAN) [23] can tackle some of the drawbacks. The number of clusters does not need to be specified a priori in DBSCAN, this is useful when the dataset is not labeled. It also means that DBSCAN could identify more arbitrary cluster shapes, since density identified similarity propagates within the dataset. Due to its density clustering properties, it is also more robust in the presence of outliers and typically has higher accuracy than k-means. Drawbacks of DBSCAN includes: ineffective distance measure for time series data; expensive to execute; and its inability to cluster data with different densities.

B. Dynamic Time Warping

Euclidean distance is a simple distance measure that is widely used with good results. However, with time series data that is continuous and possibly having small discrepancies that does not change the overall shape, Euclidean distance is not a satisfactory measure. Using a simple example from speech recognition, different people speaking the same word would convey exactly the same meaning, however this could be hard to determine using Euclidean distance since the words are likely spoken with different pitch, and at different speeds. For time series (or other sequential data), it is often the shape of data that are of more interest to us and not the specific value.

Dynamic Time Warping (DTW) is an algorithm for measuring the similarity between two temporal sequences which may vary in time or speed. Generally, this is a method that can allow computers to find an optimal match between two given sequences under certain restrictions. Its advantage is that it allows one-to-many mappings, and this allows one point to be mapped to multiple points in the other sequence. Originally, DTW was used in speech recognition, later it was adapted to various real-world data mining problems. Equations 1 and 2 shows the distance computation for Euclidean and DTW distances respectively, where given two time series sequences Q and C, $Q = q_1, q_2, ..., q_i, ..., q_n$, and $C = c_1, c_2, ..., c_j, ..., c_m$. When calculating the Euclidean distance, the total distance is the sum of the distance between each of the corresponding one-to-one mappings of q_i and c_i. In the case of DTW distance, a n-by-m distance matrix is first constructed containing the distance information between all the elements from the two sequences. The warping path is denoted as $W = w_1, w_2, ..., w_k, ..., w_K$, and while there are exponentially many warping paths, only the path minimizing *Dist(DTW)* is of interest [29].

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

$$Dist(DTW) = min\{\sqrt{\sum_{k=1}^{K} w_k}/K \tag{2}$$

Part of DTW's effectiveness is due to the algorithm "looking around" for the better mapping. There are various step patterns suited for different situations, Equation 3 can be considered a standard step pattern and will be used in conjunction with the DTW algorithm in this paper.

$$D(A_i, B_j) = \delta(a_i, b_j) + min\left\{\begin{array}{l} D(A_{i-1}, B_{j-1}) \\ D(A_i, B_{j-1}) \\ D(A_{i-1}, B_j) \end{array}\right\} \tag{3}$$

As is shown in Fig.2, the mapping results from DTW is often times more intuitive than Euclidean distance. Research continues to show that DTW to be one of the best measures so far for time series data similarity measure [6]–[14], thus for this paper we will use DTW as the distance measure for time series data, and will not further compare the differences and advantages between Euclidean distance and DTW.

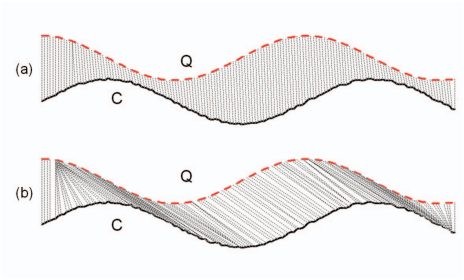

Figure 2. (a) Euclidean Mapping (top) showing one-to-one mapping and (b) DTW Mapping (bottom) showing one-to-many mapping respectively.

C. Time Series Averaging

Given the DTW warping path, there are two common methods to find the average of two time series sequences: coordinate by association, and coordinate by connected component [15]. The former is when the average sequence is calculated using the center of each mapping. Each coordinate of the average sequence will be the center of each association created by DTW [15]. The issue with this approach is that the result sequence is at least the same length as the largest of its parent sequences. When applied pairwise to a collection of time series, the length can increase substantially, and would depend on the order of calculation [25]. In the latter case, when given the warping path between two sequences, associate each connected component to a coordinate of the average time series sequence by taking the barycenter [15]. Contrary to the previous method, the average sequence length can only decrease for connected component averaging.

A technique for robust averaging template selection called the Crossword Reference Template (CWRT) was developed by Abdulla et al. to improve recognition accuracy [24]. The CWRT technique extracts a few examples from the dataset, find the sequence whose length is closest to the average length

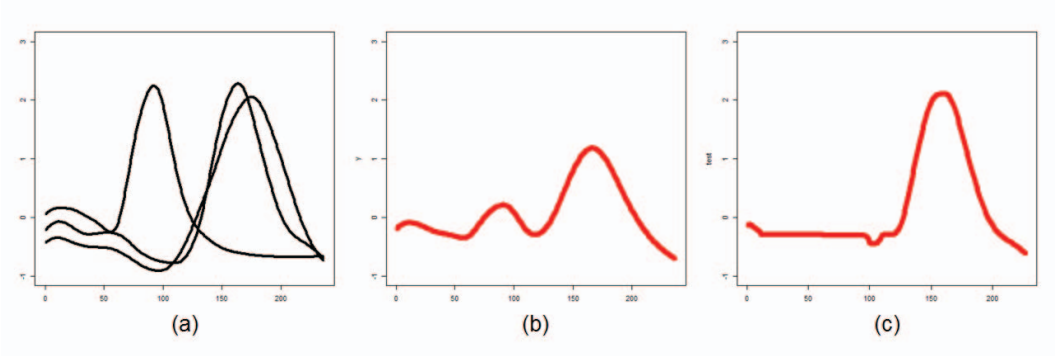

Figure 3. (a) Three out-of-phase but similar time series of the same class, (b) the average of the three time series in (a) using traditional averaging, (c) the average of the three time series in (a) using DBA.

as the initial reference template. Then other templates are aligned with the template using DTW, and the final reference template is obtained by averaging the time-aligned templates across each frame [24]. This method is invariant to the order of sequence processing, however, extracting the examples and using the closest-to-average length sequence as the initial reference template may lead to inconsistent results.

The Non-Linear Alignment and Averaging Filters (NLAAF) [26] is a tournament scheme, where averaging is applied N times for N number of sequences. The average sequence is calculated using the center of each association, and the length can grow substantially. Prioritized Shape Averaging (PSA) [27] was proposed to deal with the shortcomings of NLAAF, PSA averages two sequences whose DTW distance is minimal to other sequences, and where each connected component is associated with a coordinate of the resulting mean. However, both NLAAF and PSA are local averaging strategies, and local averaging strategies are sensitive to order, meaning if the averaging process is run again the results may change. Additionally, initial error may propagate through the entire averaging process. It should also be noted that DTW-based averaging may lead to over-fitting, and the result average sequence may become longer and longer, requiring further length reducing strategies [15].

Niennattrakul et al. discussed the result using k-means with DTW [4], because DTW has no triangular inequality property, the averaged time series may not be an actual representation. They found the bottleneck of clustering time series is a lack of good averaging method for time series data. While commonly used for statistical analysis, finding the average of a time series dataset is not an easy task. Calculating an average time series with the traditional Euclidean mean, even for equal length time series, usually, would end with an averaged time series not representing any original time series. As is shown in Fig.3, when trying to calculate the average of 3 time series sequence (3 single peaked sequences) using the regular arithmetic mean, the result is a two-peaked, flat sequence, not representative of any of the original sequences. Without a proper time series averaging mechanism, k-means and k-medoid used in conjunction with DTW does not guarantee accurate results for time series clustering [4]. Petitjean et al. proposed a global method to calculate the average of time series data, the DTW Barycenter Averaging (DBA) [15]. In the case of time series data, we can think of each time series sequence as a mass, and the average sequence being the orbiting center. And this orbiting center is refined with DTW between time series and initial average by minimizing the Within Group Sum of Squares (WGSS). Given a time series set $\mathbb{S} = \{S_1, S_2, ..., S_n\}$, the time series $C = \{c_1, c_2, ..., c_t\}$ is considered an average of $\mathbb{S}$ if it minimizes:

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

DBA is a global averaging method, which means it has no sensitivity to the order of calculation. If we could make the initialization process deterministic, then each DBA process can be repeated, and the result would not change. The advantage of using DBA with DTW is that it can stretch or compress time series in an intuitive way. This is useful because is allows us to look for patterns otherwise easily missed. There are two major steps to DBA computation: initialization and convergence. The initialization is where an element is chosen as a template, and is a randomized choice in the original DBA algorithm. Convergence is not always smooth because DTW makes nonlinear distortions, but at each iteration the inertia can only decrease [15]. The process ends when either the algorithm converges, or the maximum number of iterations is reached. The resulting average sequence is the same length as the initial template.

III. K-Medoid with Time Series Averaging

In this section we introduce our Distance and Density clustering method using DTW for distance measure, and DBA for cluster updates. A similar algorithm where k-means combined with DBA was used as a comparison method [16]. Our approach differs in four aspects, first we use a more data-centric approach where medoids are used instead of the arithmetic mean; second our k-medoid initialization is not a random selection; third, using elbow point in a distance graph we take data density into consideration, and look for

more natural clusterings; additionally, the initialization step for DBA is not random. Our problem can be segmented into the following three sections: distance clustering initialization, density-based distance graph for cluster split, and cluster center update with DBA.

A. Initialization

There are two steps where initialization can affect the reproducibility of our clustering algorithm in this paper, one is for the seed selection in the clustering algorithm and the other is for the DTW Barycenter Averaging process. Although DBA is insensitive to the order of calculation, it is sensitive to initial template, therefore DBA initialization influence the final time series average results, which in turn influences the clustering results.

Since we can only expect local optimum results, we explore the impact of deterministic initialization. The intuition behind k-means++ is the belief that spreading out the initial seeds is good, and in the case of outliers, the algorithm would readjust the clustering seeds [22]. Inspired by the k-means++ algorithm, where the first seed is randomly selected, and the other seeds are chosen based on distance. We approached the initialization process from two perspectives, starting from the perimeter (less dense) region and moving in, and starting from a dense region and moving out. By selecting the first seed by as a specific point, the entire clustering process becomes a deterministic one. To select the furthest or nearest point, we apply a majority voting system.

Table I
SIMILARITY MATRIX FOR TIME SERIES EVENTS

	a	b	c	d	e
a	0	66.2	96.0	81.6	55.2
b	66.2	0	72.4	69.2	63.3
c	96.0	72.4	0	37.9	90.2
d	81.6	69.2	37.9	0	75.9
e	55.2	63.3	90.2	75.9	0
Nearest	1	0	1	1	2
Furthest	2	0	3	0	0

Table I shows an example of a distance matrix of five time series, from top to bottom and left to right, the time series are referred to as a, b, c, d, and e. From the original dataset class label, there are two classes within the five time series data, $\{c,d\}$ and $\{a,b,e\}$. Using our majority voting strategy, the nearest time series is event e with 2 votes, and the furthest is c with 3 votes.

B. Distance Graph

After obtaining the initial seed we search for the next seed. We do this by plotting a distance graph of the initial seed to all the other time series sequences. A good place to split is where this plot shows the strongest bend which is the elbow point. This method is similar to finding the ε value (maximum radius of the neighborhood from cluster seed) in Density-Based Spatial Clustering of Application with Noise (DBSCAN). Fig.4 shows the distance plot for (a) the nearest time series event e, and (b) the furthest time series event c.

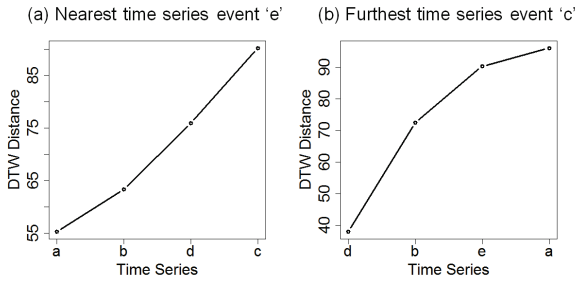

Figure 4. Distance plot of (a) nearest time series event (left) and (b) furthest time series event (right) respectively. Visually, the strongest bend is the most steep slope, which occurs between time series 'd' and 'c' in (a), and between time series 'd' and 'b' in (b).

By definition, clustering is where within cluster distances are minimized, and between cluster distances maximized; or locating the regions of high density that are separated from one another by regions of low density. Therefore, by plotting the distance to the seed in an ascending fashion and identifying the point in the plot where the bend is the strongest, we are able to find a more natural. Essentially, a strong bend in a distance plot signifies a change in virtual density among the time series dataset. In the case of the given example, the split point in Fig.4 (a) is between d and c, and in Fig.4 (b) is between d and b. Therefore the clustering result while using the nearest time series event is $\{e,a,b,d\}$, $\{c\}$; while using the furthest time series event is $\{d,c\}$, $\{b,e,a\}$. By looking at the largest DTW distance increase in the distance graph, we are able to look for more natural splits within a time series dataset. In this example, using the furthest initial seed can find the most intuitive and correct clustering result in one run, while the nearest initial seed did not cluster time series d correctly. Note this is only an example and not sufficient enough to indicate that furthest initialization is always better than nearest. With a larger dataset situations are more complicated, we cannot expect to get the final result with one split, and the clustering is also readjusted with DBA and iterated multiple times.

C. Cluster Update

Once the seeds and clusters are obtained, we update the seeds with DBA and medoids to re-balance the clusters. With the initial seed, we obtain two clusters, we treat each cluster separately and repeat the previous steps within each cluster. To avoid doubling the number of clusters each time, we choose the seed with the stronger bend from the two distance graphs as the next seed. The clusters would then be rebalanced, and the algorithm could go into the next iteration. Because the Distance Density clustering is an incremental process, the

results can be interpreted as forming a divisive hierarchy, making the results more interpretable as well as not being constrained to the initial k as is the case with k-means or k-medoids.

For a completely deterministic clustering technique, the original template used in DBA is determined using the time series sequence with the most nearest neighbor. The intuition behind this is based on the idea that time series averaged sequence is more likely to come from a dense cluster of time series. Similar to k-means or k-medoids, any meaningful help toward initialization is better than pure random. Not only will the results be reproducible, but could also potentially speed up DBA convergence.

Our Distance and Density clustering algorithm starts with a deterministic initialization, either nearest or furthest time series in the dataset. Then a distance graph is generated using the seed and based on the elbow point the dataset would be split into two clusters. A new seed will be identified in each cluster, and the distance graph with a sharper elbow point will be chosen as the next split point. This process is repeated until convergence or the maximum number of iterations is reached.

Algorithm 1 Distance Density Clustering Algorithm

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

for **do**$l \in L_{k-1}$
  $L_{k-1} \leftarrow Cluster(C_{k-1})$
  $ar[1,2,...,k-1] = DistSort(L_{k-1})$
  $value[i] \leftarrow max(arr[2]-ar[1], ..., ar[k-1]-ar[k-2])$
  if $ar[n]-ar[n-1] == max(value[i])$ **then**
    $location[i] = n$
  end if
end for
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

IV. Evaluation

In this section we demonstrate our findings and assess the performance of our Distance and Density clustering algorithm. We compare different initialization techniques in our algorithm; and show how density can improve distance clustering results. We will first introduce the datasets used in this paper, then we show the importance of clustering initialization, and finally we compare the performance of our clustering approach to traditional distance clustering.

A. Datasets

We will demonstrate our findings using the UCR time series dataset archive [29]. There are a total of 85 time series datasets in the UCR archive, including synthetic data and real-world data from various domains, ranging in class numbers as well as sizes. Each dataset consists of training and testing, because the training and testing datasets from many datasets are extremely imbalanced, we merged them for the purpose of our clustering experiments. Due to size and running time, not all datasets are included our experiments, instead we randomly chose 50.

B. Initialization

Here we compare the accuracy results from different initialization procedures, the performance will be shown using accuracy, F score, and rand index. Accuracy performance is judged by the amount of true positives and true negatives (Equation 5). F score (Equation 6) is the harmonic mean of recall and precision. F score is more resilient to class imbalance, and is a better measure than accuracy when false positives and false negatives have different cost and consequences. Rand index measures the agreements between two data clusterings, here it is the comparison between labeled data, and results obtained from our clustering algorithm.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{5}$$

$$FScore = 2 * \frac{precision * recall}{precision + recall} \tag{6}$$

The performance of different initialization techniques are shown in Fig.5 using (a) accuracy, (b) F score, and (c) rand index. The 50 datasets are labeled on the x axis, and the y axis are the measurement values. The shaded gray region is obtained by connecting the area between the best and worst performance of random initialization out of 100 runs. The orange line in each figure shows the performance of furthest seed initialization, and the blue line shows the performance of nearest seed initialization. Overall furthest sequence initialization is outperforming the rest, and nearest sequence initialization weaves through the random initialization region. There are some rough pattern where certain datasets has higher performance regardless of the initialization method or the evaluation measure, it is possible that some datasets shows stronger similarity and dissimilarity, and are easier to cluster.

Table II
When initialization is random, the number of cases where each of the two clustering method has better performance

	Distance Density Clustering	**Distance Clustering**
Accuracy	39	11
F Score	36	14
Rand Index	33	17

Fig. 5. The (a) Accuracy, (b) F Score, and (c) Rand Index of 50 datasets from the UCR repository. Using Distance Density clustering, the orange (top) line is the performance of furthest sequence initialization, the blue line is the performance of nearest sequence initialization, the gray area is obtained by shading the region between the best and worst performance of random initialization.

Table II shows the performance of our Distance Density clustering compared to distance clustering: k-means, for better adaptability to time series and a fair comparison, here we use DBA to update k-means as well. Out of the 50 datasets, there are 39 cases in accuracy, 36 cases in F score, and 33 cases in rand index where Distance Density accuracy outperforms distance clustering. The cases where Distance Density is outperforming distance clustering, it usually has better performance across all three measures, and visa versa. Therefore, it is possible that certain datasets are more adaptive to certain clustering methodology, and it would be difficult to find a methodology that is a good fit to all datasets.

V. Conclusion and Future Works

In this paper we proposed a Distance Density clustering algorithm that considers the natural splits within a dataset. We used DTW for more accurate distance description, and DBA for intuitive time series averaging and cluster seed updates. Our clustering methodology is a medoid-based clustering algorithm, but applies distance graphs to search for clusterings based on the virtual density of time series data. Our contribution can be summarized as follows: first, we proposed a furthest sequence initialization technique for time series data that has promising results for clustering, it is significant in the sense that by using deterministic initialization we can guarantee a good and stable clustering result that is reproducible; secondly, by considering the distribution of the data, we are able to improve distance-based clustering with density, without having to deal with the high time complexity of a full density-based methodology. Additionally, our heuristic clusters in a divisive hierarchical manner, which could be useful in visualization and incremental clustering.

In the future we plan to apply improved DTW algorithm to improve the speed of our algorithm. Additionally, the datasets in the UCR archive are all univariate data, whereas many real world data such as finance, natural phenomenons are multivariate, we believe it would be very useful to develop an effective method to cluster multivariate time series data.

ACKNOWLEDGMENT

This project has been supported in part by funding from CISE, MPS and GEO Directorates under NSF award #1443061, and by funding from the LWS Program, under NASA award #NNX15AF39G.

References

[1] Nagy, George. “State of the art in pattern recognition.” Proceedings of the IEEE 56.5 (1968): 836-863.

[2] Drago C, Scepi G. Time Series Clustering from High Dimensional Data[C]//International Workshop on Clustering High-Dimensional Data. Springer Berlin Heidelberg, 2012: 72-86.

[3] Jain A K. Data clustering: 50 years beyond K-means[J]. Pattern recognition letters, 2010, 31(8): 651-666.

[4] Niennattrakul V, Ratanamahatana C A. On clustering multimedia time series data using k-means and dynamic time warping[C]//Multimedia and Ubiquitous Engineering, 2007. MUE’07. International Conference on. IEEE, 2007: 733-738.

[5] Hautamaki V, Nykanen P, Franti P. Time-series clustering by approximate prototypes[C]//Pattern Recognition, 2008. ICPR 2008. 19th International Conference on. IEEE, 2008: 1-4.

[6] Sakoe H, Chiba S. A dynamic programming approach to continuous speech recognition[C]//Proceedings of the seventh international congress on acoustics. 1971, 3: 65-69.

[7] Keogh E, Ratanamahatana C A. Exact indexing of dynamic time warping[J]. Knowledge and information systems, 2005, 7(3): 358-386.

[8] Myers C, Rabiner L. A level building dynamic time warping algorithm for connected word recognition[J]. IEEE Transactions on Acoustics, Speech, and Signal Processing, 1981, 29(2): 284-297.

[9] Jeong Y S, Jeong M K, Omitaomu O A. Weighted dynamic time warping for time series classification[J]. Pattern Recognition, 2011, 44(9): 2231-2240.

[10] Berndt D J, Clifford J. Using dynamic time warping to find patterns in time series[C]//KDD workshop. 1994, 10(16): 359-370.

[11] Rakthanmanon T, Campana B, Mueen A, et al. Searching and mining trillions of time series subsequences under dynamic time warping[C]//Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2012: 262-270.

[12] Ratanamahatana C A, Keogh E. Three myths about dynamic time warping data mining[C]//Proceedings of the 2005 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2005: 506-510.

[13] Giorgino T. Computing and visualizing dynamic time warping alignments in R: the dtw package[J]. Journal of statistical Software, 2009, 31(7): 1-24.

[14] Munich M E, Perona P. Continuous dynamic time warping for translation-invariant curve alignment with applications to signature verification[C]//Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on. IEEE, 1999, 1: 108-115.

[15] Petitjean F, Ketterlin A, Ganarski P. A global averaging method for dynamic time warping, with applications to clustering[J]. Pattern Recognition, 2011, 44(3): 678-693.

[16] Paparrizos J, Gravano L. k-shape: Efficient and accurate clustering of time series[C]//Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. ACM, 2015: 1855-1870.

[17] Gusfield D. Algorithms on strings, trees and sequences: computer science and computational biology[M]. Cambridge university press, 1997.

[18] Petitjean F, Ganarski P. Summarizing a set of time series by averaging: From Steiner sequence to compact multiple alignment[J]. Theoretical Computer Science, 2012, 414(1): 76-91.

[19] Petitjean F, Forestier G, Webb G I, et al. Dynamic time warping averaging of time series allows faster and more accurate classification[C]//Data Mining (ICDM), 2014 IEEE International Conference on. IEEE, 2014: 470-479.

[20] Kaufman, Leonard, and Peter Rousseeuw. Clustering by means of medoids. North-Holland, 1987.

[21] Drineas P, Frieze A, Kannan R, et al. Clustering large graphs via the singular value decomposition[J]. Machine learning, 2004, 56(1): 9-33.

[22] Arthur D, Vassilvitskii S. k-means++: The advantages of careful seeding[C]//Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2007: 1027-1035.

[23] Ester M, Kriegel H P, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise[C]//Kdd. 1996, 96(34): 226-231.

[24] Abdulla W H, Chow D, Sin G. Cross-words reference template for DTW-based speech recognition systems[C]//TENCON 2003. Conference on Convergent Technologies for the Asia-Pacific Region. IEEE, 2003, 4: 1576-1579.

[25] Costanzo J A W B. Regularization of Dynamic Time Warping Barycenter Averaging, with Applications in Sign Classification[M]. Rochester Institute of Technology, 2013.

[26] Gupta L, Molfese D L, Tammana R, et al. Nonlinear alignment and averaging for estimating the evoked potential[J]. IEEE Transactions on Biomedical Engineering, 1996, 43(4): 348-356.

[27] Niennattrakul V, Ratanamahatana C A. Shape averaging under time warping[C]//Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on. IEEE, 2009, 2: 626-629.

[28] Chen Y, Keogh E, Hu B, et al. The ucr time series classification archive[J]. www.cs.ucr.edu/~eamonn/time_series_data, 2015.

[29] Keogh, Eamonn J., and Michael J. Pazzani. “Derivative dynamic time warping.” Proceedings of the 2001 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2001.