Density-based Clustering of Time Series Subsequences

Anne Denton
Department of Computer Science
North Dakota State University
Fargo, North Dakota 58105, USA
anne.denton@ndsu.nodak.edu

ABSTRACT

Doubts have been raised that time series subsequences can be clustered in a meaningful way. This paper introduces a kernel-density-based algorithm that detects meaningful patterns in the presence of a vast number of random-walk-like subsequences. The value of density-based algorithms for noise elimination in general has long been demonstrated. The challenge of applying such techniques to time-series data consists in first specifying uninteresting sequences that are to be considered as noise, and secondly ensuring that those uninteresting sequences will not affect the clustering result. Both problems are addressed in this paper and the success of the technique is demonstrated on several standard data sets.

1. INTRODUCTION

Frequent pattern mining algorithms are among the most important contributions of the data mining community to the data analysis toolbox. Frequent subsequences in time series data are of interest by themselves [13] and in combinations that form strong association rules [5]. Literature on association rule mining has largely assumed that time series subsequences can be encoded by letters through partitioning of the occurring subsequences using k-means clustering. This approach is motivated by its similarity to vector quantization [7] and can be justified through its property of minimizing the squared error function. It has been noted that the resulting cluster centers are, however, not very specific to the data set from which they originate [12]. This does not disqualify them from use in association rule mining since the technique does not require a data set dependent representation. It is, nevertheless important to determine if clustering can be used to mine for frequently occurring subsequences.

This paper demonstrates that kernel-density-based clustering [3, 9] is indeed capable of identifying cluster centers that are specific to the data set from which they originate. These cluster centers represent frequent subsequences and can, thereby, have applications similar to motifs [13]. The derivation based on the concept of a kernel-density has an important fundamental benefit: It will be shown in section 3 that normalization can make some sequences much more frequent than others, even for data that is created in a random fashion. It is, therefore, important to compare any result against the probability of getting that same result based on random input. The construction of a density landscape makes this comparison very easy.

Why, then, does it make a difference if density-based clustering is used instead of k-means or hierarchical clustering? Time series data often contain significant noise, i.e. subsequences that are the result of random fluctuations. Density-based clustering only considers those regions of the density-landscape as clusters that rise above a noise threshold. Setting the threshold appropriately can eliminate the impact of noise, provided the noise distribution leads to an approximately constant density surface. The noise tolerance of density-based clustering is extensively discussed in [9]. Adaptation to time series data requires a detailed understanding of what constitutes noise in that setting. Different time series models will be covered in section 2 to allow identification and targeted elimination of noise-like sequences.

The paper is organized as follows: Section 2 provides background on time series models and kernel-density-based clustering, section 3 describes the algorithm, section 4 demonstrates the results on real data, and 5 concludes the paper.

2. BACKGROUND

Many types of data are recorded as a function of time. This paper focuses on the analysis of a single sequence of real-valued data.

Definition 1: A time series $T = t_1, ..., t_n$ is a sequence of real numbers. Numbers correspond to values of an observed quantity, collected at equally spaced points in time.

Definition 2: A subsequence of time series $T = t_1, ..., t_n$, with length w, is a sequence $S = t_m, ..., t_{m+w-1}$ with $1 \leq m \leq n - w + 1$. The process of extracting subsequences by incrementing m in steps of one is called application of a sliding window. Subsequences will be represented as vectors in a w-dimensional vector space.

Clustering of any kind of data requires the definition of a similarity or of a distance measure. One of the best-known distance measures, and a popular choice in time series clustering, is the Euclidean distance. The Euclidean distance measure is a special case of an L_p norm. L_p norms may fail

to capture similarity well when being applied to raw time series data because differences in the average value and average derivative affect the total distance. Normalization is an important step to reduce this problem. In this paper Z-normalization is used in which the mean of the subsequence is subtracted and the data are divided by their standard deviation.

Other specialized distance measures have been described for time series clustering, such as dynamic time warping, DTW [2], and longest common subsequence similarity, LCSS [16]. They are not suitable to the short subsequences of interest in this paper.

2.1 Kernel-density-based Clustering

Clustering based on kernel-density estimation has been successfully used in many contexts [3, 4, 9]. Assuming n data points $x_i, i = 1, ..., n$ in a d-dimensional space, the kernel density estimator is given by

$$\hat{f}(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) \tag{1}$$

where the kernel function $K(x)$ is normalized

$$\int_{\mathbb{R}^d} K(\mathbf{x}) d\mathbf{x} = 1. \tag{2}$$

In this paper, a Gaussian kernel is used throughout

$$K^{(G)} = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{|\mathbf{x}|^2}{2}\right) \tag{3}$$

Since the Gaussian kernel is radially symmetric, a profile can be defined through

$$K(\mathbf{x}) = c_{k,d} k(|\mathbf{x}|^2) \tag{4}$$

where $c_{k,d}$ is a normalization constant that guarantees the normalization in equation (2). The goal of finding representative sequences is achieved by identifying the points in the density landscape that correspond to the highest density. In [9] these local maxima are referred to as density attractors and in [4] as modes. The modes are determined by picking starting points and updating their location through a hill climbing step. Following [4], updates, i.e. differences between tentative cluster center locations for successive steps, are computed as

$$\mathbf{m}_h(\mathbf{x}) = \frac{\sum_{n=1}^{n} \mathbf{x_i} g\left(\left|\frac{\mathbf{x}-\mathbf{x_i}}{h}\right|^2\right)}{\sum_{n=1}^{n} g\left(\left|\frac{\mathbf{x}-\mathbf{x_i}}{h}\right|^2\right)} - \mathbf{x} \tag{5}$$

where $g(x) = -k'(x)$ is the negative derivative of the kernel profile. Any data point can be associated with the cluster center to which it is attracted. Only modes above a threshold t are considered cluster centers. Data points that are attracted to modes below t are outliers or noise.

Clusters cannot only be defined based on the density attractors or modes but also as regions that are continuously above a threshold [9]. Such a definition allows multiple attractor regions to be joined into one arbitrarily-shaped cluster. In the case of time series subsequence clustering the focus lies on the identification of representative sequences and joining of attractor regions is thereby not considered helpful.

Kernel-density-based clustering is robust against noise, provided the noise leads to an approximately constant density surface. Constant contributions to the density distribution do not affect the position of maxima. The description of noise as uniformly distributed data points, which underlies the proof of noise invariance in [9], is not valid for typical time series subsequences. In the following it will be shown what distribution can be expected for time series data.

2.2 Time Series Models

Several time series models have been proposed [14]. We will now look for a model of a time series that describes noise in typical data sets well and can, thereby, serve as a noise definition.

The arguably simplest model of a time series is "strict white noise" as defined in [14], denoted by $\{e_t\}$. Mean, μ, and variance σ are assumed to be the same for all time points.

$$\begin{aligned} E\{e_t\} &= \mu, \quad \text{var}\{e_t\} = \sigma^2, \forall t, \\ &\quad \text{cov}\{e_t, e_s\} = 0, \forall t \neq s. \end{aligned} \tag{6}$$

Data that follow this distribution lead to a density landscape that is similar to that of random data in other subject domains. There will be a broad maximum at the value of μ in each dimension, which will be moved to the origin by the normalization. Kernel-density-based clustering identifies local maxima as clusters, provided they are higher than the broad maximum due to noise.

Time series data are not, however, typically described well by the model of "strict white noise". In the vast majority of settings we can expect some correlation between the values at successive points in time. For example, in stock market data the default assumption is for share values to remain constant. Nobody would consider resetting all expected share values to 0 or a constant value on a day without trading. Instead share values are assumed to approximately retain the value they had on the previous day of trading. There is correspondingly little benefit in eliminating "strict white noise". Clusters may still be determined by factors that are common to most time series and, thereby, reflect standard behavior rather than data-set-specific patterns. In this case clusterings could be similar or identical for different series and would still be considered meaningless according to [12].

A more advanced concept is the linear time series model in which future values are assumed to depend on past values as a linear superposition [14]

$$\sum_{u=0}^{\infty} h_u X_{t-u} = e_t \tag{7}$$

where the X_t are time series values, $\{h_u\}$ is a sequence of real values, and $\{e_t\}$ is a sequence of independent zero mean random variables, as defined in equation (6) with $\mu = 0$. In the simplest case, the next value of the time series depends only on one current value. A process in which future states only depend on the present state is called a Markov process. Assuming further that the expression on the left hand side of equation (7) is simply the difference between successive steps, we arrive at a time series model that qualifies as a good example of a random sequence in most settings. In

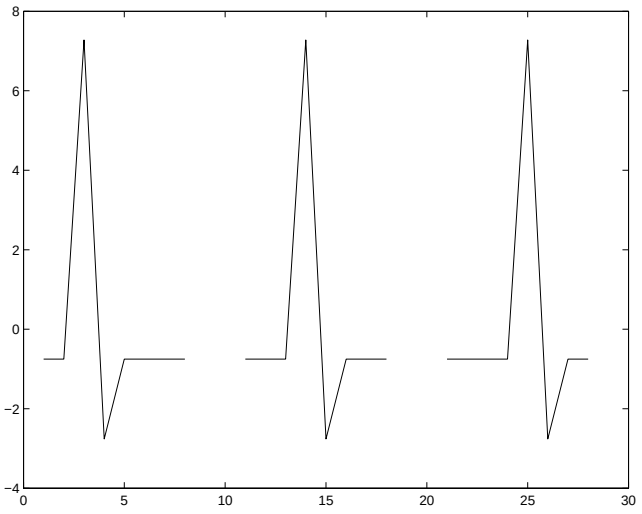

Figure 1: Subsequences in which a characteristic pattern occurs in different locations.

stock data, for example, random fluctuations would neither give us reason to buy nor to sell stock. Deviations from such behavior, on the other hand, are considered interesting patterns. Such a time series is also called a random walk time series.

2.3 Random Walk Time Series

The concept of a random walk was introduced to describe particle motion in n dimensions [15]. For time series data time advances regularly and the random walk is restricted to one dimension, corresponding to the values the time series can take.

Definition 3: A Gaussian random walk time series is a normally distributed sequence that satisfies

$$\begin{aligned} &X_t - X_{t-1} = e_t \\ &E\{e_t\} = 0, \quad \text{var}\{e_t\} = 1, \ \forall t, \\ &\text{cov}\{e_t, e_s\} = 0, \ \forall t \neq s. \end{aligned} \tag{8}$$

The random walk time series that is used as an example series to evaluate the effectiveness of the algorithm was taken from [11] and follows this model. Note that the choice of mean and variance are irrelevant in this definition because subsequences are normalized.

A slightly modified version of this concept is used to derive a random walk space that can be completely enumerated. It is then necessary to limit the number of possible random values to a finite number, in this case 1 and -1. Such a simplification is also common for random walks in their traditional use [15].

Definition 4: A discrete random walk time series is a sequence that satisfies

$$\begin{aligned} &X_t - X_{t-1} = r_t \\ &E\{r_t\} = 0, \quad |r_t| = 1, \forall t, \\ &\text{cov}\{r_t, r_s\} = 0, \ \forall t \neq s. \end{aligned} \tag{9}$$

Definition 5: The space of all discrete random walk time series of a given length is called random walk space.

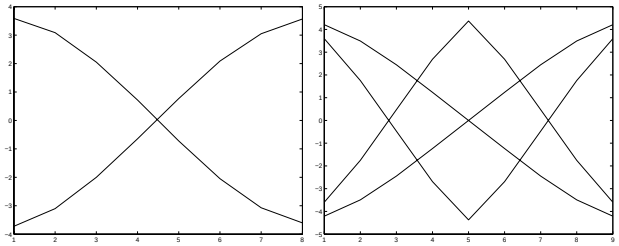

Figure 2: Density-based clustering of a Gaussian random walk (left) and a persistent discrete random walk (right).

Finally, it can be observed that for many time series not only the values of successive points are correlated, but also the slopes between them, i.e., if there is an increase between time $t-1$ and time t there is more likely to be an increase than a decrease from t to $t+1$. Random walks with such correlations are called persistent. While not all time series are persistent, those that are, should be treated with an appropriate noise model. Patterns that are entirely due to persistence are not expected to be of interest to a user because they are independent of the specific data set. The same patterns would be observed for all data sets with the same level of persistence.

Definition 6: A persistent random walk time series is a sequence that satisfies

$$\begin{aligned} &X_t - X_{t-1} = r_t \\ &E\{r_t\} = 0, \quad |r_t| = 1, \ \forall t, \\ &\text{cov}\{r_{t-1}, r_t\} = c, \ \forall t. \end{aligned} \tag{10}$$

where c is a constant that is determined from the data. If the time series of interest does not show persistence, this model reduces to equation (9).

3. APPLICATION OF KERNEL-DENSITY-BASED CLUSTERING TO TIME SERIES

Kernel-density-based clustering can be implemented in two fundamentally different ways. Data points can be stored in a table that is parsed once for each iteration of the hill-climbing algorithm [3, 4]. Alternatively a grid representation can be used within the space that is spanned by all attributes [9]. In the latter approach the volume of the space increases exponentially with the number of dimensions of the data. The former approach was therefore taken to achieve acceptable scaling to large subsequences. The implementation was done in MATLAB as an extension of [10], a kd-tree-based [1] MATLAB toolbox for kernel-density estimation. Modes are evaluated using a hill-climbing algorithm. Starting points are all data points that are larger than their d nearest neighbors within the data set, where d is the number of dimensions (points in each subsequence). This strategy was tested against the more rigorous choice of picking each data point as starting position. Results showed that all modes were reliably identified and the speed was significantly increased by limiting the set of starting points. Comparison of each point with its neighbors is very fast given the kd-tree representation.

Further modifications were necessary to adapt the density-

```
 1  length = 8 % length of subsequences
 2  sigma = 2.1% variance of Gaussian in Kernel Density Estimation
 3  threshold = 0.3e-4 % density threshold
    % Prepare subsequences
 4  dataArray = normalize(slidingWindow(sequence))
    % Prepare random walk representation
 5  rwLibrary= createRandomWalks(length)
 6  rwArray = closestRandomWalk(dataArray,rwLibrary)
 7  ratio = successiveSlopesSameVsDifferent(rwArray)
 8  rwKernelDensity = densityOfRws(length,sigma,ratio)
    % Calculate weights to compensate for random walk density
 9  rwKernelDensityMatrix = evaluate(rwKernelDensity,rwLibrary)
10  weights = conjugateGradient(rwKernelDensityMatrix,vectorOfOnes)
11  weights = applyLowerBound(weights,lowerBound)
12  weightArray = mapToData(rwArray,weights)
    % Actual clustering step using kernel density estimation
13  dataKernelDensity = kde(dataArray,sigma,weightArray)
14  start = getMaxima(dataKernelDensity,length)
15  [clusterCenters,centerDensities] = modes(dataKernelDensity,start)
16  clusterCenters = filterCloseCenters(clusterCenters)
17  clusterCenters = applyThreshold(clusterCenters,centerDensities,
    threshold)
```

Figure 3: Pseudocode

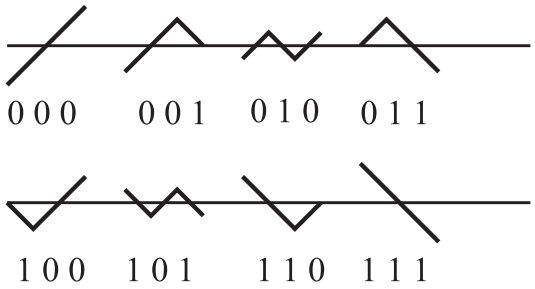

Figure 4: Random walk space for a time series of length 4.

based clustering algorithm to time series subsequences. One problem is that a characteristic pattern that is shorter than the window size will result in several cluster centers. Figure 1. illustrates the problem. A clear pattern of a large data point followed by a small one can be identified in all three sequences. The sliding window approach samples data points multiple times and can thereby lead to multiple representations of the same pattern from the original time series. A user is likely to only be interested in this pattern once. The problem was resolved as a post-processing step in which potential cluster centers are compared and ones that are closer to each other than a threshold are eliminated. Sequences are shifted and only those parts compared that are defined for both subsequences. The resulting sequences are normalized and the distance weighted (using a weight of $\sqrt{(}i+1)$ for a shift of i) to give penalty to shifted matches. Shifts up to half the length of the time series subsequence are considered.

Kernel-density-based clustering, as described so far, leads to non-trivial maxima in the density landscape even if the input sequences are perfect random walks. Figure 2. shows the result of density-based clustering of a Gaussian random walk (left) and a persistent discrete random walk (right). The next section will show how to compensate for this uninformative result.

3.1 Density Distribution of a Random Walk and its Inversion

The idea of the algorithm is as follows: The density landscape of discrete random walk data is evaluated. Input data is then weighted such that random walk data would lead to a constant density surface. Handling this problem computationally requires that only a finite number of sequences are used to define the random walk space. The discrete random walk definition in equation (9) is used for this purpose. All random walks that are possible according to this definition can be constructed in a straight forward way. Figure 4. shows the space of random walks for sequences of length 4. For any two successive data points there is a choice of an increase or decrease by 1 (3 choices for 4 data points). In general, the number of discrete random walks that have to be considered for sequences of length w is $2^{(w-1)}$. Each possible walk can be efficiently represented by a $(w-1)$-digit binary number that serves as an index for array representations of density and other properties.

Subsequences of real data, naturally, do not match these sequences exactly. Two possible prescriptions of identifying corresponding random walks for a given data sequence were evaluated. Each normalized data sequence can be compared with all possible normalized random walk sequences and the closest chosen to represent the sequence. Alternatively a difference-based matching process can be used that considers only the sign of the slope between neighboring points in the sequence. The latter approach shows better scaling to large sequences but does not capture the relevant features of the data sequences as well. The first approach was, therefore,

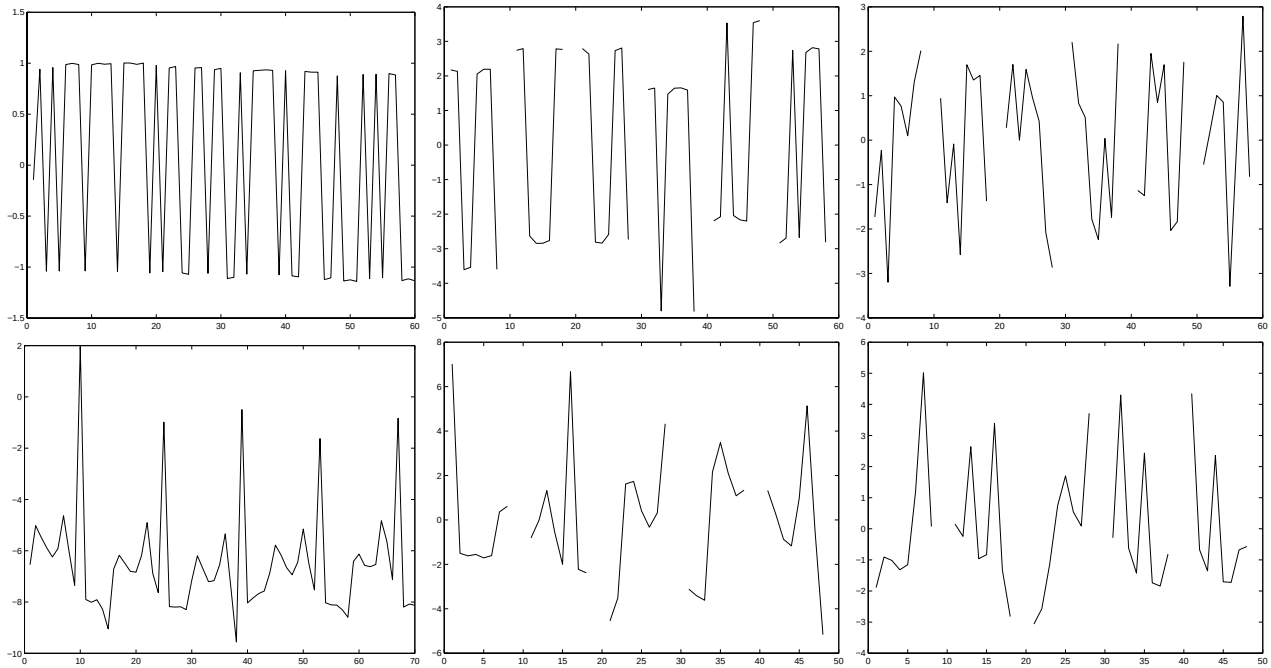

Figure 5: Data set glassfurnace (top) and ecg (bottom); shown are the time series itself (left), results of density-based clustering (middle), and results of k-means clustering (right).

taken.

Persistence, as defined in equation (10), is incorporated based on the statistics of the data. After representative random walk sequences have been picked, the number of occurrences of equal slopes is determined. The ratio of equal over different slopes is calculated. Based on that ratio, all random walks in the random walk space are weighted. A density landscape is constructed with the same kernel function as is used for the data.

The goal of calculating the density landscape of random walks is to compensate for any patterns that occur by random sequences alone. Weights are determined for each random walk model sequence in the random walk space. For each real sequence the weight of the corresponding model sequence is supplied as part of the construction of the density landscape. Weights are defined such that the weighted density landscape based on the set of model sequences is 1 for the location of each sequence. Determining the weights amounts to solving the following equation for $\mathbf{x}$

$$\mathbf{1} = \mathbf{K}^{(G)}\,\mathbf{x} \tag{11}$$

where $\mathbf{1}$ is a vector with all elements equal to one, and $\mathbf{K}^{(\mathbf{G})}$ is the kernel matrix for a Gaussian kernel

$$K_{i,j}^{(G)} = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{2}\right) \tag{12}$$

evaluated at the positions of each model sequence, i.e., each point in the random walk space. In principle, the matrix $\mathbf{K}^{(\mathbf{G})}$ could be inverted. Since equation (11) only has to be solved once, and an approximate solution is satisfactory, a biconjugate gradient method as provided in MATLAB (bicgstab) is used instead.

The resulting vector $\mathbf{x}$ can – and commonly does – show negative values. This result is not surprising since equation (11) is not guaranteed to have a solution, for which all elements of $\mathbf{x}$ are positive. In practice it is, however, questionable if input sequences should be weighted with a negative weight. Doing so would mean that the existence of a particular sequence in the data set leads to a decrease in the density landscape. The minimum acceptable value for elements of $\mathbf{x}$ was set to be 0.01 times the maximum value occurring in $\mathbf{x}$. All smaller values were set to that minimum value.

3.2 Summary of the Algorithm

Figure 3. summarizes the steps in the algorithm. As a first step subsequences are extracted using a sliding window and are normalized, step 4. Several different normalization techniques were evaluated such as using the derivative of sequences as discussed in [6]. As a further alternative the original subsequences were taken, the mean subtracted and the standard deviation of differences between successive points was used for normalization, compare equation (8). Over all, Z-normalization led to the best performance and was chosen.

The next steps 5.-12. deal with the creation of a random walk representation, steps 5.-8., and the creation and approximate inversion of the kernel matrix, steps 9.-12. These steps are specific to the time series subsequence setting and are not used in the standard and adapted density-based clustering variants in section 4.

The clustering steps 13.-16. are implemented as outlined in section 2.1. Note that this algorithm may still return maxima as a result of noise in the form of random sequences. Due to the somewhat artificial nature of the reference sequences as discrete persistent random walks, Gaussian ran-

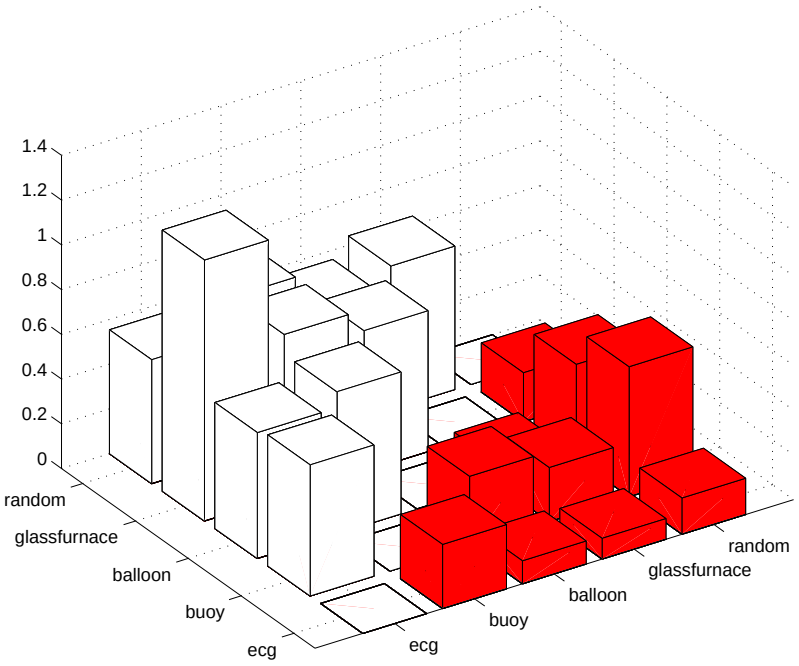

Figure 6: Comparison of density-based (dark bars, right half) with k-means clustering (white bars, left half).

dom walks will still result in systematic maxima similar to the left part of figure 2., albeit with a much lower density. The lower bound on weights also contributed an uneven density surface for random walks. Density values of clusters that were evaluated based on Gaussian random data were used as a guideline to determine a suitable threshold, below which maxima are not considered as cluster centers. Step 17. ensures that only clusters that have a density higher than that threshold are returned.

4. EXPERIMENTAL EVALUATION

The algorithm was evaluated on several standard data sets from the UCR Time Series Data Mining Archive [11] as well as on an ecg series (MIT-BIH Arrhythmia Database: mitdb100) from PhysioBank [8]. The ecg series was compressed by averaging over 20 consecutive values, the buoy series from the UCR Archive by averaging over 4 values. All experiments were done using subsequences of length 8. The width of the Gaussian kernel was chosen to be 2.1 and the density threshold for cluster centers 0.3×10^4.

Figure 5. (top) shows a data set from [11] together with the clustering results from density-based and k-means clustering. The cluster centers from density-based clustering clearly represent the typical patterns in the time series. Cluster centers from k-means clustering, on the other hand, bear no resemblance to the original data. One may argue that this time series has a rather extreme shape. Figure 5. (bottom) shows the same comparison for ecg data. Although the contrast is not as extreme, it can again be clearly seen that most of the k-means cluster centers do not represent any particular part of the sequence. No parts of that ecg sequence show two similarly high, pointed maxima next to each other. Yet, three of the k-means clusters that show such subsequences. All density-based clusters can be identified with subsequences of the original time series.

4.1 Meaningfulness of Clusterings

Following the idea in [12] the meaningulness of clustering is evaluated. For this purpose data sets were broken in half, and the cluster centers derived from both halves were compared using the meaningfulness-measure

$$\begin{aligned} &\text{cluster\_distance\_n(A, B)} \\ &\quad \equiv \frac{1}{k_A} \sum_{i=1}^{k_A} \min \left[dist(\overline{a}\_i, \overline{b}\_j) \right], \quad 1 \leq j \leq k_B \end{aligned} \tag{13}$$

$$\begin{aligned} &\text{clusteringmeaningfulness(X, Y)} \\ &\quad \equiv \frac{\text{within\_set\_X\_distance\_n}}{\text{between\_set\_X\_and\_Y\_distance\_n}} \end{aligned} \tag{14}$$

where $A = (\overline{a}_1, \overline{a}_2, ..., \overline{a}_{k_A})$ are the cluster centers derived from the first half of the data set and $B = (\overline{b}_1, \overline{b}_2, ..., \overline{b}_{k_B})$ are the cluster centers derived from the second half. This test is stricter than the original test that was based on separate runs of the k-means algorithm on the same data. Since density-based clustering has no random component the original test could not be applied. Since clusterings could differ in the number of clusters the cluster_distance measure had to be normalized with the number of sequences.

The average number of clusters in density-based clustering, averaged over all data sets listed in table 3.2., was approximately 4 (4.3). Comparisons are, therefore, done with k-means clustering, with $k = 4$. Only data sets were used that showed cluster centers with densities higher than clusters in random data. This criterion excluded some popular data

Table 1: Meaningfulness Results for Different Combinations of Data Sets

	mean (excl. random)		buoy	balloon	glassfurnace	random
Weighted Density-based Clustering	0.189	ecg	0.29	0.10	0.09	0.16
		buoy		0.31	0.24	0.58
		balloon			0.10	0.42
		glassfurnace				0.22
Adapted Density-based Clustering	0.228	ecg	0.26	0.15	0.11	0.08
		buoy		0.52	0.15	0.63
		balloon			0.18	0.29
		glassfurnace				0.06
Standard Density-based Clustering	0.390	ecg	0.18	0.21	0.46	0.10
		buoy		0.47	0.41	0.63
		balloon			0.60	0.36
		glassfurnace				0.34
K-means Clustering	0.718	ecg	0.59	0.56	1.17	0.55
		buoy		0.63	0.72	0.72
		balloon			0.63	0.56
		glassfurnace				0.64

sets used in other works such as the Standard and Poor 500 stock index. With the current sensitivity of the algorithm it is not possible to distinguish between the stock index and random walk data.

Figure 6. shows the results of density-based clustering with compensation for a persistent random walk density distribution (right, dark) together with results from k-means clustering (left, white). It can clearly be seen that the density-based results lead to much smaller values corresponding to more meaningful results.

It can also be seen that the difference between k-means and density-based clustering is not as significant for comparisons with random walk data. This is expected since the similarity between runs on different random walk sequences should not be similar. That means that the within_set_X_distance_n for random data should be as high as the between_set_X_and_Y _distance_n to other sequences. A meaningfulness value of about 0.5 should, therefore, be expected for any calculation involving random data. The fact that meaningfulness values are rather lower is an indication that cluster centers for random data still do have components of the systematic results in figure 2. because the model was built on discrete rather then continuous random walks, and weights were restricted to be above a threshold.

Table 3.2. shows detailed results for the runs in fig. 6. as well as for two other variants of density-based clustering: Standard density-based clustering uses the modes finding algorithm from [10] without modifications, and adapted density-based clustering allows shifting of subsequences in comparisons (see section 2.3).

5. CONCLUSIONS

Kernel-density-based clustering was successfully adapted to time series subsequence data. It was, thereby, shown that time series subsequence clustering can lead to meaningful results. A random walk space was introduced as a means of defining a reference distribution of sequences. Persistence of random walks was included in the treatment. The reference distribution was then used to derive weights for time series subsequence data. These weights compensated for the uneven distribution of random walk data. The resulting algorithm was evaluated on several standard data sets.

6. REFERENCES

[1] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9), 1975.

[2] D. Berndt and J. Clifford. *Advances in knowledge discovery and data mining*, chapter Finding patterns in time series: a dynamic programming approach, pages 229–248. AAAI Press, Menlo Park, CA, 1996.

[3] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[4] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[5] G. Das, K.-I. Lin, H. Mannila, G. Renganathan, and P. Smyth. Rule discovery from time series. In *Proceedings of the IEEE Int. Conf. on Data Mining*, Rio de Janeiro, Brazil, 1998.

[6] M. Gavrilov, D. Anguelov, P. Indyk, and R. Motwani. Mining the stock market (extended abstract): which measure is best? In *Sixth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 487–496, Boston, MA, 2000.

[7] A. Gersho and R. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, Boston, MA, 1992.

[8] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000 (June 13).

[9] A. Hinneburg and D. Keim. A general approach to clustering in large databases with noise. *Knowl. Inf. Syst.*, 5(4):387–415, 2003.

[10] A. Ihler. Kernel density estimation toolbox for matlab (r13), accessed 04/2003.
[11] E. Keogh and T. Folias. The ucr time series data mining archive, 2002.
[12] E. Keogh, J. Lin, and W. Truppel. Clustering of time series subsequences is meaningless: implications for previous and future research. In *Proceedings of the IEEE Int. Conf. on Data Mining*, pages 115–122, Melbourne, FL, 2003.
[13] P. Patel, E. Keogh, J. Lin, and S. Lonardi. Mining motifs in massive time series databases. In *Proceedings of the IEEE Int. Conf. on Data Mining*, Maebashi City, Japan, 2002.
[14] M. Priestley. *Non-linear and non-stationary time series analysis*. Academic Press, 1988.
[15] F. Reif. *Fundamentals of Statistical and Thermal Physics*. McGraw-Hill, New York, NY, 1965.
[16] M. Vlachos, D. Gunopoulos, and G. Kollios. Discovering similar multidimensional trajectories. In *Proceedings 18th International Conference on Data Engineering (ICDE'02)*, San Jose, CA, 2002.