J Intell Inf Syst (2012) 39:287–315
DOI 10.1007/s10844-012-0196-5

Rotation-invariant similarity in time series using bag-of-patterns representation

Jessica Lin · Rohan Khade · Yuan Li

Received: 14 July 2011 / Revised: 19 December 2011 / Accepted: 31 January 2012 /
Published online: 1 April 2012

Abstract For more than a decade, time series similarity search has been given a great deal of attention by data mining researchers. As a result, many time series representations and distance measures have been proposed. However, most existing work on time series similarity search relies on shape-based similarity matching. While some of the existing approaches work well for short time series data, they typically fail to produce satisfactory results when the sequence is long. For long sequences, it is more appropriate to consider the similarity based on the higher-level structures. In this work, we present a histogram-based representation for time series data, similar to the "bag of words" approach that is widely accepted by the text mining and information retrieval communities. We performed extensive experiments and show that our approach outperforms the leading existing methods in clustering, classification, and anomaly detection on dozens of real datasets. We further demonstrate that the representation allows rotation-invariant matching in shape datasets.

Keywords Time series · Similarity search · Feature extraction · Representation · Classification · Structural similarity

1 Introduction

Time series similarity search has been a major research topic for time series data mining for the past two decades. As a result, many time series representations and

J. Lin (✉) · R. Khade · Y. Li
Computer Science Department, George Mason University, Fairfax, VA, USA
e-mail: jessica@cs.gmu.edu

R. Khade
e-mail: rkhade@gmu.edu

Y. Li
e-mail: ylif@gmu.edu

similarity measures have been proposed (Chan and Fu 1999; Faloutsos et al. 1994; Keogh 2002; Keogh and Kasetty 2002; Keogh et al. 2001; Li and Vitanyi 1997; Lin et al. 2007). Similarity measures can be categorized, based on how features are extracted and how similarity is determined, into shape-based similarity and structure-based similarity (Keogh 2004). The former determines the similarity of two time series by comparing their individual point values, whereas the latter looks at the higher-level structures (Keogh et al. 2004). Most existing approaches focus on finding shape-based similarity. Classic examples of shape-based similarity measures for time series include the Euclidean Distance and Dynamic Time Warping (Keogh 2002). While some of these approaches work well for short time series data, they typically fail to produce satisfactory results with long time series data (Keogh 2002). To understand the need for a higher-level, structure-based similarity measure for long time series data, consider the scenario for text data. If we are to compare two strings, we can use the string edit distance to compute their similarity. However, if we want to compare two documents, we typically do not compare them on the word-to-word basis. Instead, it is more meaningful to use a higher-level representation that can capture the structure or semantic of the document. Below, we describe the two types of similarity in more detail.

Given two time series Q and C, shape-based similarity determines how similar these two datasets are by summing up local comparisons. The most well-known distance measure in data mining literature is the Euclidean distance, for which sequences are aligned in the point-to-point fashion, i.e. the ith point in sequence Q is matched with the ith point in sequence C. Assuming that Q and C are of the same length n, Eq. 1 defines their Euclidean distance.

$$D(Q, C) \equiv \sqrt{\sum_{i=1}^{n} (q_i - c_i)^2} \tag{1}$$

The simplicity and efficiency of Euclidean distance makes it the most popular distance measure in data mining, and it has the advantage of being a distance metric. While Euclidean distance works well in general, it requires that both input sequences be of the same length. In addition, it is sensitive to distortions, e.g. shifting along the time axis. As an example, the top and bottom time series in Fig. 1 appear to be very similar. In fact, the time series below is the shifted version of the time series above. However, the slight shifts along the time axis will result in a large distance between the two time series.

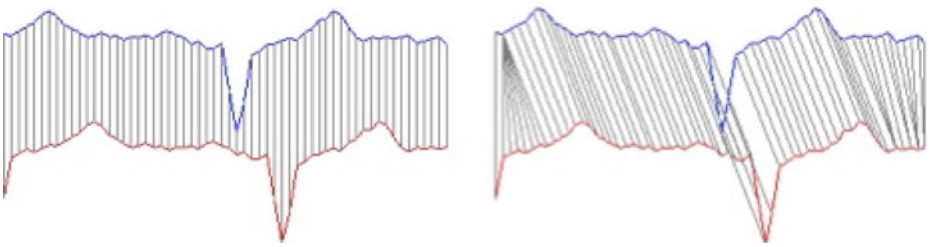

Fig. 1 (*Left*) Alignment for Euclidean distance between two time series. (*Right*) Alignment for Dynamic Time Warping distance between two time series

Such a problem can generally be handled by elastic distance measures such as Dynamic Time Warping (DTW) (Keogh 2002; Ratanamahatana and Keogh 2004; Sart et al. 2010; Ko et al. 2005). DTW uses dynamic programming to determine the best alignment that will produce the optimal distance. The parameter, warping window length, determines how much warping is allowed to find the best alignment (Ratanamahatana and Keogh 2004). A large warping window causes the search to become prohibitively expensive, as well as possibly allowing meaningless matching between points that are far apart. On the other hand, a small window might prevent us from finding the best solution. Euclidean distance can be seen as a special case of DTW, where there is no warping allowed. Figure 1 demonstrates the difference between the two distance measures. Note that with Euclidean distance, the dips and peaks in the two time series are misaligned and therefore not matched, whereas with DTW, they are aligned with their corresponding points from the other time series. While DTW is a more robust distance measure than Euclidean Distance, it is also a lot more computationally intensive. Keogh (2002) proposed an indexing scheme for DTW that allows faster retrieval. Nevertheless, DTW is still at least several orders slower than Euclidean distance.

Other elastic distance measures exist for time series, such as Longest Common SubSequence (LCSS) (Vlachos et al. 2002), Edit distance with Real Penalty (ERP) (Chen and Ng 2004) and Time-Warp Edit Distance (TWED) (Marteau 2009). However, we do not consider them in our experimental comparison since DTW is still considered the most commonly used distance measure among the family of elastic measures for time series (Ding et al. 2008). In addition, it performs just as well, if not better, than the other elastic measures (Ding et al. 2008).

Shape-based similarities work well for short time series; however, for long time series data (generally speaking, with a length of hundreds or more), they either produce poor results, or require intensive computation (Keogh et al. 2004). More specifically, while Euclidean distance is efficient to compute, it is very sensitive to distortions such as noise and shifting. On the other hand, while DTW is more robust, its computational time is quadratic in the lengths of the input time series. Recently, it has been proposed that a more appropriate alternative to determining similarity between long and noisy time series is to measure their similarity based on higher-level structural information (Keogh et al. 2004). Compared to the large amount of work on shape-based similarity, there is relatively little work on finding structural similarity. This is an unfortunate oversight, as the structural approach is particularly useful for long time series, or for applications where domain experts compare time series or signals based on the arrangement of morphological events present in the signals (Olszewski 2001), e.g., radar signal detection and speech recognition. Several structure- or model-based similarities have been proposed that extract global features such as autocorrelation, skewness, and model parameters from data (Keogh 2004; Nanopoulos et al. 2001; Wang et al. 2006). However, it is not trivial how to determine relevant features, and/or compute distances given these features. In addition, in many cases, it was shown that the simple Euclidean Distance outperforms many of the sophisticated model-based approaches (Keogh 2004).

In this paper, we propose a novel time series representation that allows us to determine structural (dis)similarities between time series data. Our method is robust and efficient, and it is inspired by the well-known bag-of-words representation for text data. There are several advantages for our approach compared to existing

structure-based methods. First, since the overall representation is built by extracting the subsequences from data, we in fact take local structures into consideration as well as global structures. Furthermore, the incremental construction of the representation suggests that it can be used in the streaming data scenario. Our representation also allows users to understand the pattern distribution of the data by examining the resulting histograms. It also has the desirable property of being invariant to distortions like rotation. We show that our approach outperforms leading existing methods in the tasks of classification, clustering, and anomaly detection on several real datasets. We demonstrate the rotation-invariance property with a case study on shape matching.

The rest of the paper is organized as follows. In Section 2 we briefly discuss background and related work. Section 3 presents our methodology. In Section 4, we show empirical results in clustering, classification, and anomaly detection. In Section 5, we performed a case study on rotation-invariant shape matching. We conclude and discuss future work in Section 6.

2 Background and related work

In this section, we briefly discuss background and related work on time series similarity search.

For concreteness, we begin with a definition of time series:

Definition 1 *Time Series*: A time series $T = t_1, \ldots, t_m$ is an ordered set of m real-valued variables.

Some distance measure $Dist(C,Q)$ needs to be defined in order to determine the similarity between time series objects.

Definition 2 *Distance*: *Dist* is a function that has C and Q as inputs and returns a nonnegative value R, which is said to be the distance from Q to C.

Each time series is normalized to have a mean of zero and a standard deviation of one before the distances are computed, since it is well understood that in virtually all settings, it is meaningless to compare time series with different offsets and amplitudes (Keogh and Kasetty 2002).

As mentioned, Euclidean distance and Dynamic Time Warping are among the most commonly used distance measures for time series. For this reason, we will use Euclidean distance as the distance measure for our new representation, and compare the results with Euclidean distance and Dynamic Time Warping on raw data. Since our representation is inspired by the bag-of-words representation, for which cosine similarity is used instead of Euclidean distance, we will also compare these two distance measures on our representation. In addition to comparing with well-known distance measures on the *raw* data, we also demonstrate that our method outperforms popular time series representations such as Discrete Fourier Transform (DFT).

While there have been dozens of representations and distance measures proposed for shape-based similarity matching, there is relatively little work on finding structure-based similarity. Deng et al. (1997) proposed learning ARMA model on

the time series, and using the model coefficients as the feature. This approach has an obvious limitation on the characteristics of input data. Ge and Smyth (2000) proposed a deformable Markov Model template for temporal pattern matching, in which the data is converted to a piecewise linear model. However, this approach requires many parameters, and does not achieve better accuracy than Euclidean distance (Keogh 2004). Nanopoulos et al. (2001) proposed extracting statistical features of time series such as skewness, mean, variance, and kurtosis, and classifying the data using multi-layer perceptron (MLP) neural network. Kriegel et al. (2008) proposed to approximate each time series by the coefficients of a mathematical model involving a set of reference time series. Their approach requires derivation of a set of reference time series in order to build the best fitting model.

Keogh et al. (2004) proposed a compression-based distance measure that compares the co-compressibility between datasets. Motivated by Kolmogorov Complexity (Keogh et al. 2004; Li and Vitanyi 1997) and promising results shown in similar work in bioinformatics and computational theory, the authors devised a new dissimilarity measure called CDM (Compression-based Dissimilarity Measure). Given two datasets (strings) x and y, their compression-based dissimilarity measure can be formulated as follows:

$$CDM(x, y) = \frac{C(xy)}{C(x) + C(y)} \tag{2}$$

where $C(xy)$ is the compressed size of the concatenated string $x + y$, $C(x)$ and $C(y)$ are the compressed sizes of the string x and y, respectively. In their paper, the authors show superior results compared to other existing structural similarity approaches. In this work, we will compare our method with CDM, the best structure-based (dis)similarity measure reported. We will show that our approach is highly competitive, with several additional advantages over existing methods.

3 Finding structural similarity

We propose a histogram-based similarity measure, using a representation similar to the one widely used for text data. In the Vector Space Model (Salton et al. 1975), each document can be represented as a vector. Each dimension of the vector corresponds to one word in the vocabulary, and its value is the relative frequency of occurrences for the corresponding word in the document. As a result, a p-by-q term-to-document matrix X is constructed, where p is the number of unique terms in the text collection, q is the number of documents, and each element $X(i, j)$ is the frequency of the ith word occurring in the jth document.

This "bag of words" representation is widely accepted for documents. It is able to capture the structure or topic(s) of a document, without knowing the exact locations or orderings of the word appearances. We hypothesize that we may be able to represent time series data in a similar fashion, i.e. as a combination of patterns from a finite set of patterns.

There are two challenges if we represent time series data as a "bag of patterns." The first challenge concerns with the definition and construction of the patterns "vocabulary." The second challenge comes from the fact that time series data are composed of consecutive data points. There is no clear "delimiters" between patterns. To tackle both challenges, we use a simple sliding window scheme, and adapt

the widely used symbolic representation for time series, SAX (Symbolic Aggregate approXimation) (Lin et al. 2007). The intuition is to convert the time series into a set of SAX words, and then construct a word-sequence matrix (analogous to the term-document matrix) using these SAX words. Intuitively, each SAX word represents a pattern in the time series. In the next section, we briefly describe how SAX converts a time series into strings.

3.1 Symbolic aggregate approximation

Given a time series T of length n, SAX produces a lower dimensional representation of a time series by transforming the original data into symbolic words. Two parameters are used to specify the size of the alphabet to use (i.e. α) and the size of the words to produce (i.e. w). More specifically, SAX performs the discretization by dividing a time series into w equal-sized segments. For each segment, the mean value for the points within that segment is computed. Aggregating these w coefficients forms the Piecewise Aggregate Approximation (PAA) representation of T. Each coefficient is then mapped to a symbol according to a set of breakpoints that divide the distribution space into α equiprobable regions, where α is the alphabet size specified by the user. If the symbols were not equiprobable, some of the symbols would occur more frequently than others. As a consequence, we would inject a probabilistic bias in the process. It has been noted that some data structures such as suffix trees produce optimal results when the symbols are of equiprobability (Crochemore et al. 1994). In previous work, we found that normalized time series subsequences had a highly Gaussian distribution (Lin et al. 2007). A symbolic transformation table could be created by defining breakpoints that would result in regions of equal-probability on the Gaussian distribution. These breakpoints (or the z-values) may be determined by looking them up in a statistical table. Table 1 gives the breakpoints for values of α from 3 to 10 (though higher values of α can be easily determined). For example, if $\alpha = 3$ (i.e. there are three regions), then the breakpoints that divide the distributional space into 3 equiprobable regions are -0.43 and 0.43 (see also Fig. 2 for an example).

Figure 2 shows how a time series of length 128 is converted to a SAX string *cbccbaab* (i.e. $\alpha = 3$, w = 8). The breakpoints, illustrated by dotted lines, are -0.43 and 0.43.

Table 1 A lookup table that contains the breakpoints that divides a Gaussian distribution into an arbitrary number (from 3 to 10) of equiprobable regions

a	3	4	5	6	7	8	9	10
β_1	−0.43	−0.67	−0.84	−0.97	−1.07	−1.15	−1.22	−1.28
β_2	0.43	0	−0.25	−0.43	−0.57	−0.67	−0.76	−0.84
β_3		0.67	0.25	0	−0.18	−0.32	−0.43	−0.52
β_4			0.84	0.43	0.18	0	−0.14	−0.25
β_5				0.97	0.57	0.32	0.14	0
β_6					1.07	0.67	0.43	0.25
β_7						1.15	0.76	0.52
β_8							1.22	0.84
β_9								1.28

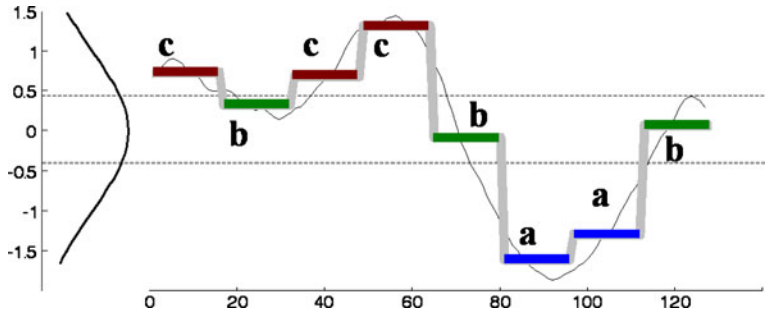

Fig. 2 Example of SAX for a time series. The time series above is transformed to the string *cbccbaab*, and the dimensionality is reduced from 128 to 8

3.2 Bag-of-patterns representation for time series

Having discretized the time series via SAX, our algorithm works as follows. First, we construct the pattern "vocabulary" for our time series database. The easiest way to achieve this is to use a sliding window to extract subsequences of length n (a user-defined parameter). Each subsequence is normalized to have a mean of *zero* and standard deviation of *one* before it is converted to a SAX string. As a result, we obtain a set of strings, each of which corresponds to a subsequence in the time series. As noted in (Lin et al. 2007), given a subsequence S_i, it is likely to be very similar to its neighboring subsequences, S_{i-1} and S_{i+1} (i.e. those that start one point to the left, and one point to the right of S_i, respectively), especially if S_i is in the smooth region of the time series. As a result, in some cases we might see that multiple consecutive subsequences are mapped to the same string. These subsequences are called trivial matches of S_i. To avoid over-counting these trivial matches as true patterns, a technique called *numerosity reduction* is often administered (Lin et al. 2007). More specifically, we may choose to record only the first occurrence of the string, and ignore the rest until we encounter a string that is different. In other words, for each group of consecutive identical strings, we may record only the first occurrence and count this group of occurrences only once. As an example, suppose we obtain the following sequence of SAX strings with the sliding window technique:

$$S = \text{aac aac abc abb abb abb abb bac baa ...}$$

With the numerosity reduction option, we would record the following sequence instead:

$$S1 = \text{aac}_1\ \text{abc}_3\ \text{abb}_4\ \text{bac}_8\ \text{baa}_9$$

The subscripts denote the starting offsets of the first occurrences of the repeating strings.

While the inclusion of numerosity reduction seems reasonable here, we believe that the decision on adapting numerosity reduction depends on the data. More specifically, for data that are generally smooth and slow changing, we might see many consecutive subsequences being mapped to the same SAX string. In this case,

counting all occurrences of the string would result in over-counting of the pattern. Figure 3 illustrates such an example. In the figure, first few extracted subsequences are shown. Since this time series is relatively smooth (i.e. no rapidly changing patterns, spikes, or noises), the first few subsequences are very similar and are likely to be mapped to the same SAX string. It would make sense to count this pattern (represented by the same SAX string) only once, rather than multiple times, until we see a different string that breaks the consecutive identical streak of strings.

Once we obtain the set of strings for each time series, we can construct the word-sequence matrix. Given α and w as the parameters for SAX, we know the size of our entire collection of possible SAX strings, or our "dictionary." More specifically, there are α^w possible SAX words. For example, for $\alpha = 4$ and $w = 4$, our dictionary size is only 256. Clearly, the size of the dictionary increases exponentially with the increase of w. Typically, a value of 3 or 4 works well for most time series datasets (Lin et al. 2007). In most experiments, we choose $\alpha = 4$ for simplicity. With this arbitrarily chosen parameter, we want to demonstrate that in situations where we have no prior knowledge on the datasets and the best parameter setting, the "default" value α works reasonably well.

Having fixed α, we now have to determine the value for w and n (the sliding window length). While the best choices of w and n are data-dependent, in general, time series with smooth patterns can be described with a small w and a large n, and those with rapidly changing patterns may prefer large w and small n to capture the critical changes. We choose $w = 6 \sim 8$ for our initial experiments, with sliding window length of $100 \sim 300$ depending on the characteristics of the data. If training data is available, these parameters can also be learned, as we will show in the later part of our experiments.

With $\alpha = 4$ and $w = 8$, the resulting dictionary size is $\alpha^w = 4^8 = 65{,}536$. Despite its apparent size, the matrix is likely to be sparse, as with text data. In our experiments, we find that only about 10% of all strings have some subsequence mapped to them. Therefore, we can eliminate words that never occur in any data, or store only the list of occurring SAX strings for each time series. Another option is to set a threshold for the maximum allowable dictionary size, and set the parameters accordingly. We will discuss this approach in a later section.

The construction of the temporal "bag of patterns" matrix M is straightforward. The matrix M is a word-sequence matrix, analogous to term-document matrices used for information retrieval and text mining. Each row i denotes a SAX word (i.e. a pattern) from the pattern dictionary; each column j denotes a time series dataset; and each $M_{i,j}$ stores the frequency of word i occurring in time series j. The matrix

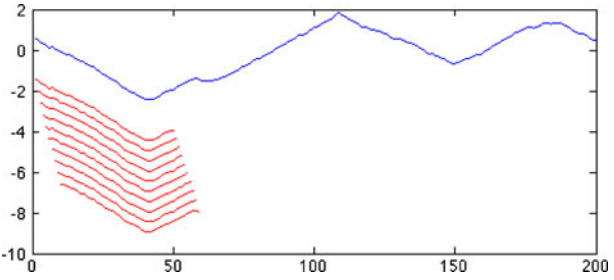

Fig. 3 Subsequences are extracted from the time series via a sliding window. The first few subsequences are very similar, so they are likely to be mapped to the same string

provides a summary of time series data in terms of distributional structure for the patterns. Once we build the matrix M, we can then use any applicable distance measures, typically Euclidean distance, or dimensionality reduction techniques to compute the similarity between them. In our experiments, we normalize the histograms so that the values range between zero and one. Doing so allows us to compare time series of different lengths meaningfully. We call this new representation BOP (Bag of Patterns).

Figure 4 shows a visual example of this representation. Like the bag-of-words representation for documents, the orderings of patterns are lost. However, for long time series data, this level of details is exactly the reason why conventional shaped-based approaches do not work well. In addition, the extraction of *subsequences* rather than *points* allows some preservation of pattern ordering within a subsequence. As our experiments demonstrate, BOP produces very good results even without knowing the exact ordering of the patterns.

Figure 5 shows one of the datasets we use in our experiments, and their corresponding "bags of patterns." The leftmost column shows thirteen pairs of time series obtained from UCR Time Series Archive (Keogh et al. 2006a) and PhysioNet (Goldberger et al. 1997). The middle column shows the bag of patterns for each time series (i.e. the column vectors in matrix M). The SAX parameters used here are $\alpha = 4$, and $w = 6$, so the resulting vocabulary has $4^6 = 4{,}096$ words (i.e. the length of the column vectors is 4,096). The numbers to the right are the sparsity levels of the vectors, calculated by the following equation:

$$s = 1 - \frac{\#of\ nonzero\ entries}{vocabulary\ size} \tag{3}$$

We can make the following observations from the sparsity levels: (1) the matrix is extremely sparse, similar to the case for text data. The densities for the bags of patterns range from 0.46% to 9%. (2) Data from the same cluster have similar distributional information and sparsity levels. The latter explains why using structural or pattern distributional information to distinguish data is a promising idea.

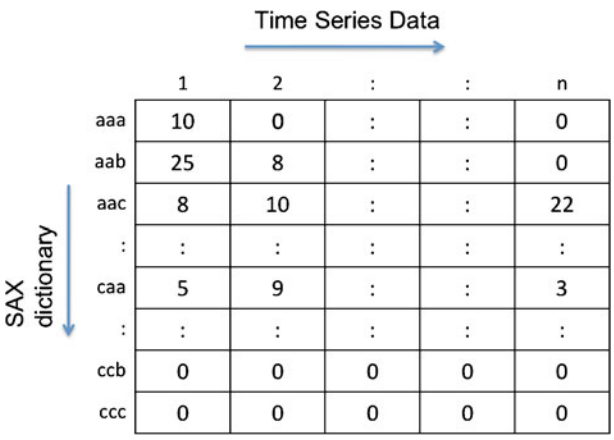

Fig. 4 A visual example of the bag-of-patterns representation for time series. Each row denotes a SAX word, and each column denotes a time series data

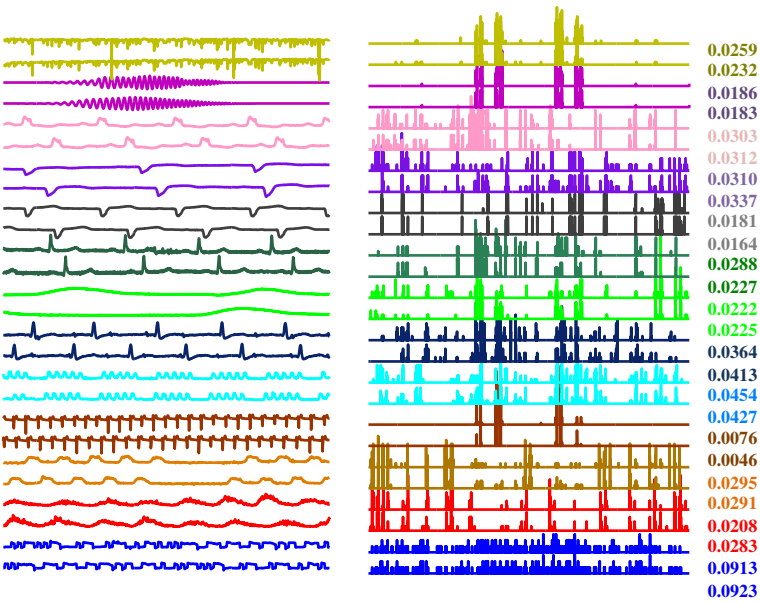

Fig. 5 Twenty-six datasets and their corresponding bags of patterns. The sparsity scores on the right show that the matrix is extremely sparse

4 Empirical evaluation[1]

In this section, we present empirical evaluation of our method on clustering, classification and anomaly detection.

4.1 Clustering

For this part of experiments, we demonstrate the effectiveness of our approach in hierarchical clustering and partitional clustering. We show that our representation outperforms leading existing approaches and produces more accurate clustering results.

4.1.1 Hierarchical clustering

One of the most widely used clustering approaches is hierarchical clustering (Johnson 1967). Hierarchical clustering computes pairwise distances of the objects (or groups of objects) and produces a nested hierarchy of the clusters. It has several advantages over other clustering methods. More specifically, it offers great visualization power with the hierarchy of clusters, and it requires no input parameters. However, its intensive computational complexity makes it infeasible for large datasets.

[1] All datasets used in this paper are available at http://www.cs.gmu.edu/∼jessica/BOP

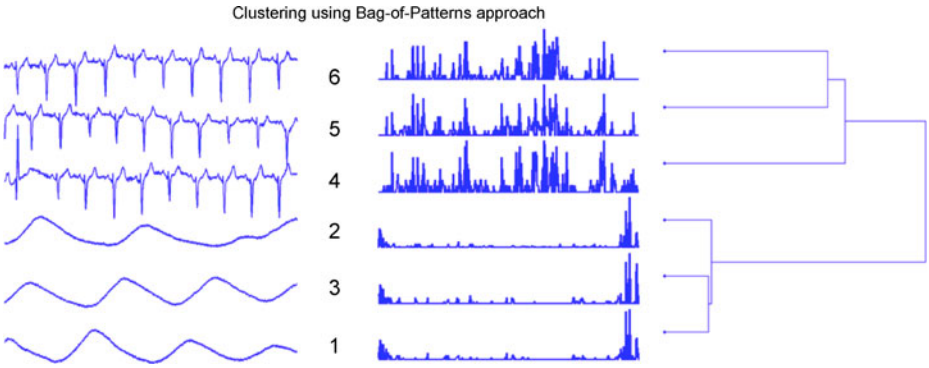

Fig. 6 New clustering result on the same data shown in Fig. 2. This time, we used our bag-of-patterns approach, and combined it with Euclidean distance. The two clusters are well separated

To give a visual intuition of why our method works, we performed a simple experiment. We extracted subsequences of length 2,048 from six different records on PhysioNet (Goldberger et al. 1997), an online medical archive containing digital recordings of physiological signals. Signals labeled #1 $\sim$ 3 are measurements on respiratory rates, and signals labeled #4 $\sim$ 6 are electrocardiograms (ECGs). All datasets were normalized prior to clustering so that they have a mean of zero and a standard deviation of one.[2] We extracted the bag of patterns from the data, and used Euclidean distance to compute the similarity between the pattern frequency vectors (i.e. the column vectors). Figure 6 shows the resulting dendrogram. Note we are now clustering on the transformed time series, or the "histogram" of the patterns. The parameters used to produce this dendrogram are $n = 200$, w $= 8$, and a $= 4$; group average linkage was used (though other linkage options produced similar results). For clarity, we also plot the original, corresponding time series to the left of the dendrogram. This figure provides the exact explanation on the clustering result, as we can see clearly that the time series clustered together have similar pattern distribution.

We also tried to cluster the raw signals using Euclidean distance as the distance measure, and the result is disappointing. Figure 7 shows the hierarchical clustering result using Euclidean distance and single linkage (the choice of linkage technique also does not make any difference in this case). As the figure shows, none of the signals was clustered correctly. One reason for the poor clustering result could be that the signals within the same cluster are not perfectly aligned. For example, signals #1 and #2 are out of phase. In addition, the presence of anomalous points, e.g., in the beginning of signal #4, could also throw off the distances computed.

[2]In previous work (Lin and Li 2009), we used a different normalization technique, min-max normalization, which resulted in slightly better clustering for Euclidean distance: signals #5 and #6 were correctly clustered together. However, z-normalization is more commonly used for time series data mining tasks.

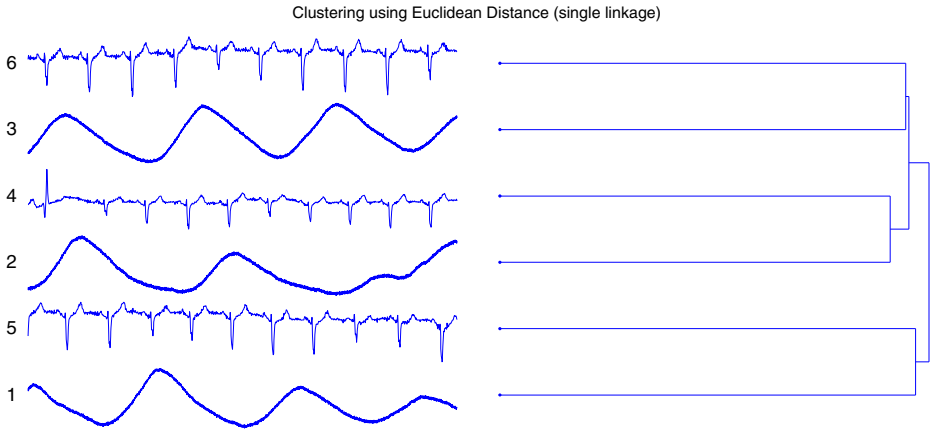

Fig. 7 Result of hierarchical clustering using Euclidean distance on raw data. None of the signals was clustered correctly

Dynamic Time Warping can be used to mitigate the out-of-phase problem. Indeed, we were able to obtain perfect clustering result using DTW. However, we would like to emphasize that DTW is computationally expensive – it is a quadratic algorithm whereas the bag-of-patterns approach is linear.

As expected, CDM produced just as good clustering result as the bag-of-patterns and as DTW did. However, the repeated saving of data to the disk makes it less efficient compared to the bag-of-patterns technique.

For our next experiment on hierarchical clustering, we used the time series datasets shown in Fig. 4. The data, obtained from different domains, have diverse structures. The datasets contain: winding (1 & 2), sunspot (3 & 4), power consumption (5 & 6; 9 & 10), ECG (7 & 8; 11 & 12; 15 & 16; 17 & 18; 19 & 20), respiratory (13 &14), shuttle (21 & 22), flutter (23 & 24), and balloon (25 & 26). Although our method does not require the input time series to have the same length, the raw Euclidean distance does. Thus we keep each dataset at length 1000. We compared our approach with the following methods: (1) Euclidean distance on raw time series ("raw Euclidean Distance"), (2) DTW on raw time series, and (3) Euclidean distance on DFT coefficients.

Figure 8 shows the dendrogram produced by hierarchical clustering using the raw Euclidean distance. Since each class consists of a pair of time series, we can easily check at the loweset-level of the dendrogram the branches separating pairs of time series. We adapt the metric proposed by Keogh et al. (2004), and compute Q, which is the number of correct birufications divided by thirteen, the number of pairs in the data. The Q-value ranges between 0 and 1, where a Q-value of 1 denotes perfect clustering. We can see that out of thirteen pairs of datasets, only three pairs are successfully clustered together. The Q-value is thus 0.23. As illustrated in Fig. 9, DTW shows some improvement over Euclidean distance and correctly clusters all but two pairs of data (i.e. 3 & 4, 21 & 22). The Q-value is 0.85 for DTW.

Next, we compare our representation with another well-known time series representation, Discrete Fourier Transform (DFT) (Agrawal et al. 1993). DFT approximates the signal with a linear combination of basis functions, and its coefficients

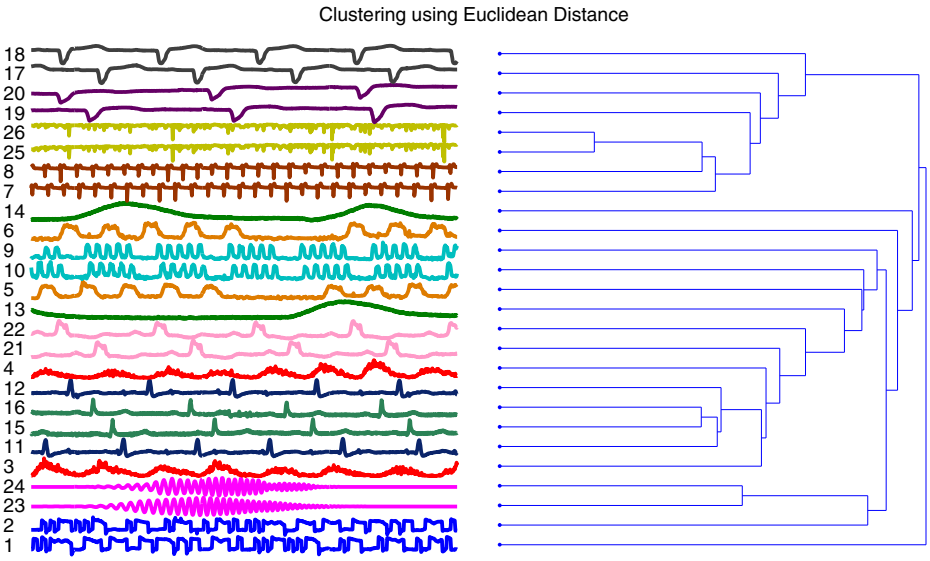

Fig. 8 Clustering result using Euclidean distance on the raw data. Only three pairs of data are cleanly clustered together (15 & 16; 23 & 24; 25 & 26)

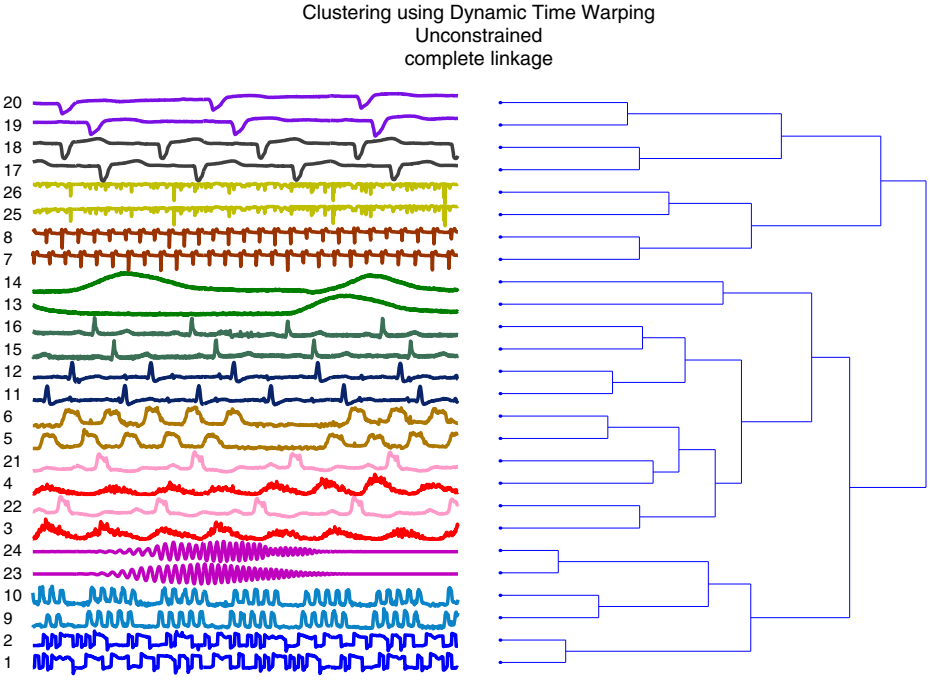

Fig. 9 Clustering result using Dynamic Time Warping on the raw data. All but two pairs of time series (3 & 4; 21 & 22) are clustered correctly

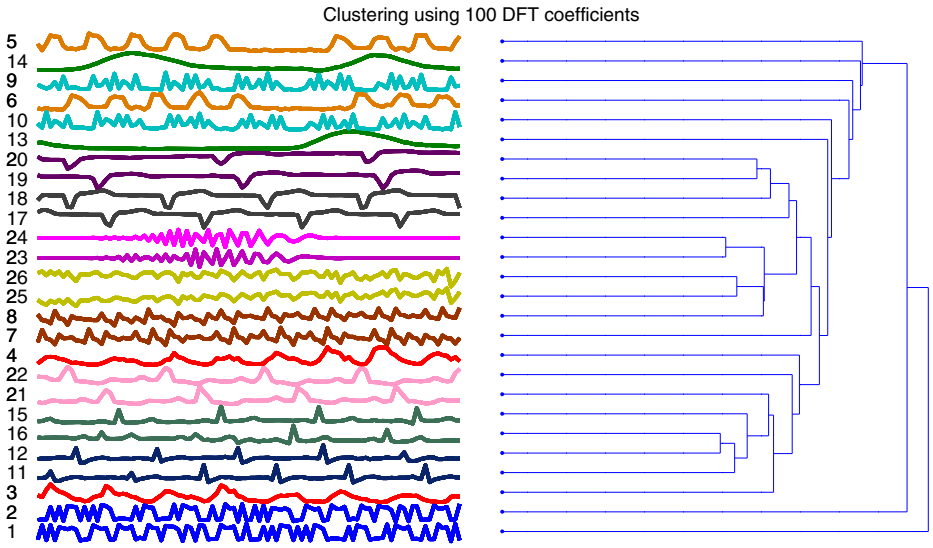

Fig. 10 Clustering results using 100 DFT coefficients and Euclidean distance. Four pairs of data are correctly clustered (17 & 18; 19 & 20; 23 & 24; 25 & 26)

represent global contribution of the signal. One of the advantages of DFT is that it offers dimensionality reduction. As demonstrated by Agrawal et al. (1993), most "energy" concentrates on the first few DFT coefficients. Therefore, we can use only a few DFT coefficients to approximate the data, while still preserving the general shape of the data. If we use all the coefficients, then we get the equivalence of the original sequence. In this experiment, we used 100 coefficients (compared to 1,000 data points in the raw data). Similar to the raw Euclidean distance, only four pairs of data are correctly clustered. Figure 10 shows the result. The Q-value for DFT is 0.31.

Figure 11 shows the clustering result produced by our Bag-of-Patterns approach. Again, the histograms for the patterns are shown in the middle of the figure. The parameters used are $n = 100$, w $= 6$, $\alpha = 4$, with no numerosity reduction, and with complete linkage for clustering. However, it turns out that the option for numerosity reduction and the choice of linkage in this experiment have no effect in clustering accuracy. As the figure shows, all thirteen pairs are correctly clustered. The Q-value for the Bag-of-Patterns representation is 1. The Q-values for the various techniques are summarized in Table 2.

The results above are promising. However, these time series are still relatively short, and they have very diverse structures. To see how our algorithm handles very long sequences, and on data with less diverse structures, we performed hierarchical clustering on the ECG dataset presented by Keogh et al. (2004). This dataset, which we will call ECG1, contains 20 ECG records that form 4 clusters. Details on the datasets can be found in the paper (Keogh et al. 2004). Each record is of length 15,000. The parameters we used are $n = 300$, $w = 6$, $\alpha = 4$. The clustering result for CDM can be found in (Keogh et al. 2004). Our results are comparable to that of CDM – both methods were able to correctly identify all four clusters, regardless of the high level of noises and misalignments in the data (see Fig. 12).

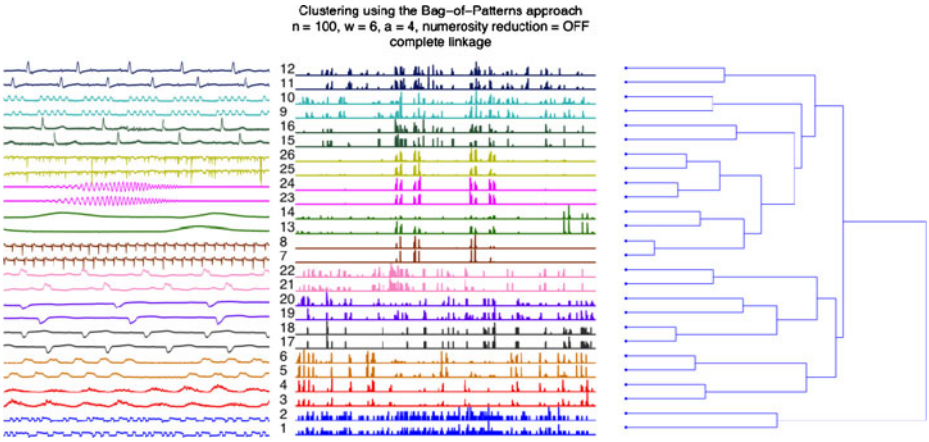

Fig. 11 Clustering results using our approach. All pairs of data are successfully clustered

To measure the hierarchical clustering results quantitatively, one way is to draw a line vertically on the dendrogram so that the line partitions the data into k clusters ($k = 4$ in this example). We can then determine the cluster membership by looking at the data grouped under each node that intersects with the line. Figure 12 shows an example. Each "block" of data (shaded/non-shaded) denotes one cluster, with a total of four clusters. Objects in each cluster share a common node. Once we obtain the cluster membership, we can evaluate clustering quality using any existing clustering validity measures. More specifically, we compare our cluster labels with the true labels, and compute the clustering quality using the evaluation method proposed by Gavrilov et al. (2000). The evaluation method compares the similarity between two sets of cluster labels, and returns a number between 0 and 1 denoting how similar the sets of clusters are. Ideally, we would like the number to be as close to 1 as possible. For the Bag-of-Patterns approach, the clustering accuracy is 1 (i.e. perfect clustering).

We compare our results with three other methods: Euclidean distance on the raw data, DTW on the raw data, and Euclidean distance on DFT coefficients. The results are shown in Figs. 13, 14, and 15, respectively. Clustering accuracy for all techniques compared is summarized in Table 3.

For the final comparison, we converted the time series to DFT coefficients, and clustered the data on the coefficients. We tried different resolutions (100–1,000 coefficients, or 0.67%–6.67% of data), but obtained poor results regardless of the resolution. The result shown in Fig. 15 uses 1,000 DFT coefficients.

While CDM produces similar results (see Keogh et al. 2004 for the dendrogram), our approach offers several advantages. With our approach, we cluster on the pattern histograms. This allows us to see the distribution of patterns from these pattern his-

Table 2 Clustering quality in Q-value for Euclidean distance, DTW, DFT, and the Bag-of-Patterns

	Euclidean distance	DTW	DFT	Bag-of-Patterns
Q-values	0.23	0.85	0.31	1

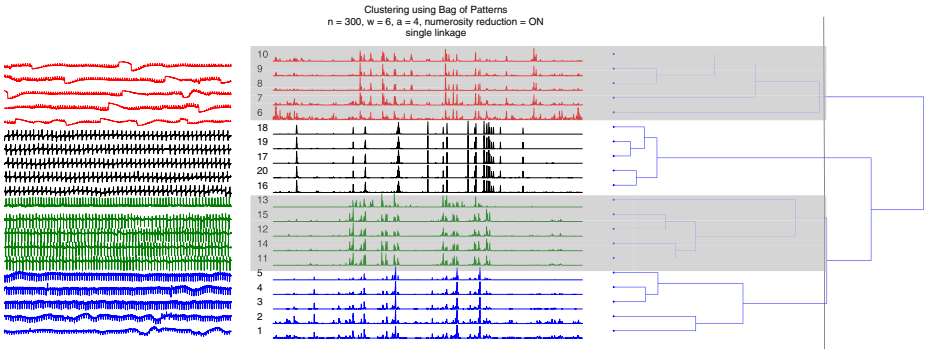

Fig. 12 Clustering result on 20 ECG datasets, using our bag-of-patterns approach. Each record is 15,000 points long. The clusters (according to the partitions indicated by the vertical line) are: cluster1: {6, 7, 8, 9, 10}, cluster2: {16, 17, 18, 19, 20}, cluster3: {11, 12, 13, 14, 15}, cluster4: {1, 2, 3, 4, 5}. The clustering accuracy is 1

tograms, and gain insights on the underlying structures of the data. Furthermore, the Bag-of-Patterns representation is fast to compute and has a linear time complexity.

4.1.2 Partitional clustering

Although the visualization power of hierarchical clustering provides a sanity check for our representation, it has limited utility due to its poor scalability. The most commonly used data mining clustering algorithm is *k*-means (Bradley et al. 1998; Lin et al. 2004; McQueen 1967). The basic intuition behind *k*-means (and in general, iterative refinement algorithms) is the continuous reassignment of objects into different clusters, so that the intra-cluster distance is minimized.

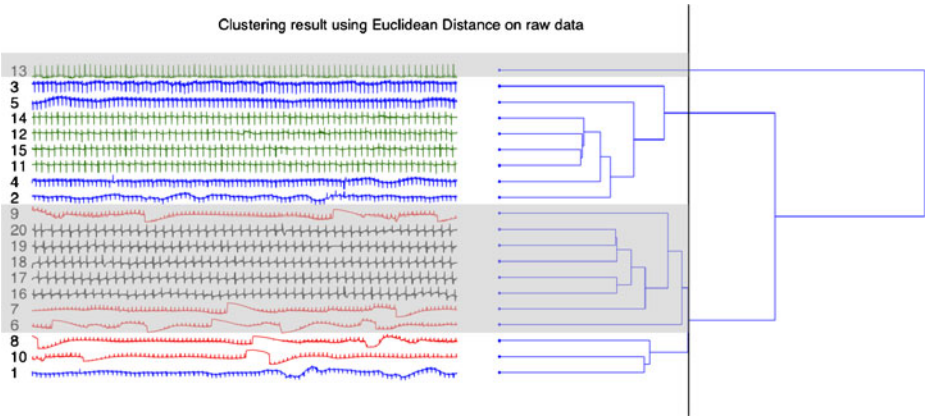

Fig. 13 Clustering result on raw ECG1 data using Euclidean Distance. The clusters (according to the partitions indicated by the vertical line) are: cluster1: {13}, cluster2: {2, 3, 4, 5, 11, 12, 14, 15}, cluster3: {6, 7, 9, 16, 17, 18, 29, 20}, cluster4: {1, 8, 10}. The clustering accuracy is 0.56

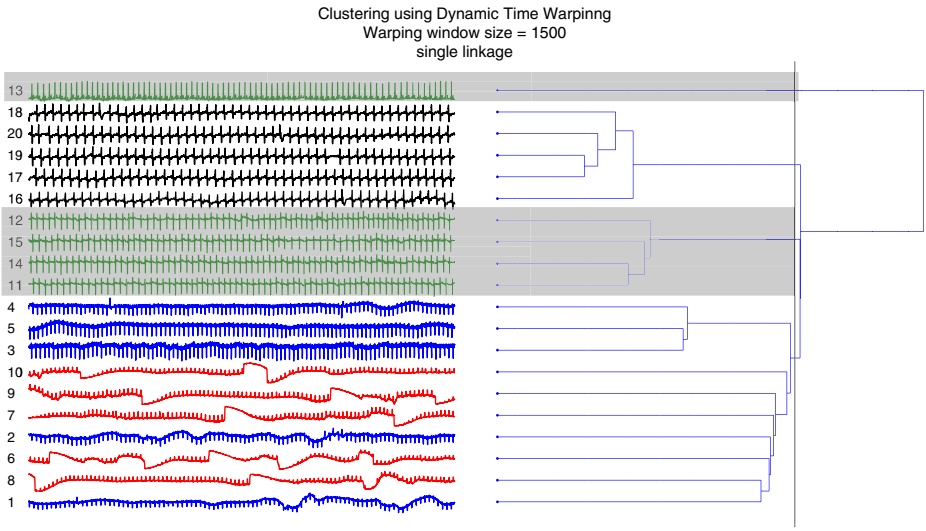

Fig. 14 Clustering result on ECG2 using Dynamic Time Warping. The clusters (according to the partitions indicated by the vertical line) are: cluster1: {13}, cluster2: {16, 17, 18, 19, 20}, cluster3: {11, 12, 14, 15}, cluster4: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}. The clustering accuracy is 0.72

We performed *k*-means using the Euclidean distance on (1) the raw data, and (2) the bag-of-patterns representation. CDM is not included in this experiment, as it's unclear how to define the centroid of a cluster (Keogh et al. 2004).

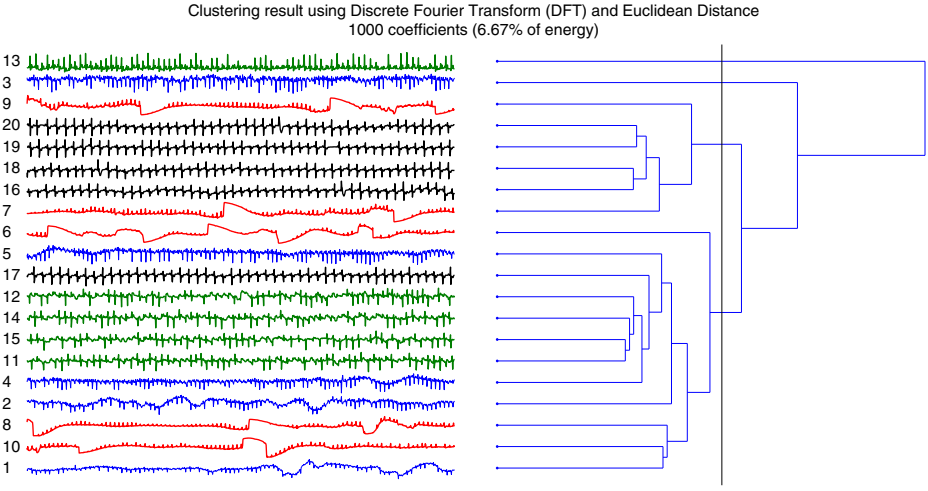

Fig. 15 Clustering result on ECG1 data using 1000 DFT coefficients (6.7% of the time series). The clusters (according to the partitions indicated by the vertical line) are: cluster1: {13}, cluster2: {3}, cluster3: {7, 9, 16, 18, 19, 20}, cluster4: {1, 2, 4, 5, 6, 8, 10, 11, 12, 14, 15, 17}. The clustering accuracy is 0.47

Table 3 Clustering quality in Q-value for Euclidean distance, DTW, DFT, and the Bag-of-Patterns

	Euclidean distance	DTW	DFT	CDM	Bag-of-Patterns
Clustering accuracy	0.56	0.72	0.47	1	1

For this experiment, we extracted 250 records from the PhysioNet archive. Each record contains 2,048 points. These records are extracted from various databases containing different vital signs, or patients with different heart conditions. We separated the records into 5 classes, and labeled them according to the databases that they are extracted from. We will call this dataset ECG2. Figure 16 shows one example from each of the 5 classes in ECG2 dataset.

We ran k-means algorithm 30 times, and recorded the clustering labels obtained from the run with the smallest objective function (i.e. sum of intra-cluster distances). We then compare our cluster labels with the true labels, and compute the clustering quality using the evaluation method proposed by Gavrilov et al. (2000). Our approach achieves the best clustering quality (0.71 vs. 0.46). The parameters we used are $n = 160$, $w = 4$, $\alpha = 6$. The results are shown in Table 4 (the row labeled "*k-means*").

4.2 Classification

Classification of time series has attracted much interest from the data mining community (Geurts 2001; Keogh and Kasetty 2002; Nanopoulos et al. 2001; Ratanamahatana and Keogh 2004; Radovanovic et al. 2010; Wei and Keogh 2006; Ye and Keogh 2009; Mueen et al. 2011; Xing et al. 2011). For the classification experiments, we will consider the most common classification algorithm, the nearest neighbor classification.

4.2.1 ECG2 dataset

To demonstrate the effectiveness on 1-nearest-neighbor classification, we used the same ECG2 dataset described in the previous section. We used the leave-one-out cross validation, and counted the number of correctly classified objects, *cc*. The accuracy is the ratio of *cc* and the total number of objects (i.e. 250). The parameters used are the same as those for the clustering experiments, i.e. $n = 160$, $w = 4$,

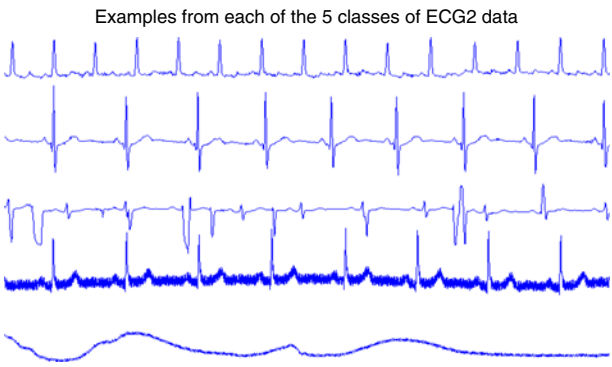

Fig. 16 One example from each of the 5 classes in ECG2 dataset

Table 4 Accuracy of our approach on clustering, classification, and discord discovery on ECG2 compared to other methods. Our approach achieves the best accuracy for all tasks. All numbers are between 0 and 1

	Euclidean distance	DTW (ww = 200)	CDM	Bag-of-Patterns
k-means	0.46	N/A	N/A	0.71
1-NN	0.44	0.73	0.69	0.996
Discord	0.35	N/A	N/A	0.85

$\alpha = 6$. For this experiment, we also added DTW and CDM. We used the same SAX parameters for CDM as the Bag-of-Patterns method. We make no claim that this is the *best* performance for CDM, as the parameters were chosen to match ours (which were arbitrary choices) rather than trained. Better selection of parameters might have resulted in better accuracy for CDM (or our method). The accuracy results are shown on the second row, labeled 1-NN, in Table 4. The improvement over the other methods is astounding. For our approach, the accuracy of 0.996 means that there is only 1 misclassified object, out of 250 objects.

4.2.2 Gun datasets

In this experiment, we repeated the experiments shown on the Gun dataset used in Keogh et al. (2004). The Gun dataset consists of four time series extracted from video sequences in which two actors with and without a replica gun perform a series of actions: draw the gun; aim at the target; and return it to the holster. The four time series represent the following: (A) Actor 1 with gun; (B) Actor 1 without gun; (C) Actor 2 with gun; and (D) Actor 2 without gun. We followed the same experimental settings described by Keogh et al. (2004), and extracted 20 random subsequences of length 1,000 from each of the sequences, obtaining a total of 80 sequences of length 1,000. This was divided into 2 classification problems. The first is a 4-class problem that classifies the four acts, and the second is a 2-class problem that differentiates between the two actors. More details on the Gun dataset can be found in Keogh et al. (2004).

We ran 1-NN classification using the leave one out approach for validation using Euclidian Distance, DTW, CDM, and Bag-of-Patterns. The dataset was z-normalized prior to discretization (for CDM and Bag-of-Patterns) or classification. We used SAX parameters $n = 128$, $n = 16$ and $\alpha = 3$, with numerosity reduction for both CDM and Bag-of-Patterns. The results are given below. The 2-class problem is relatively easy, and all techniques achieved very good results. However, for the 4-class problem, Bag-of-Patterns is the only one that achieves perfect accuracy (Table 5).

Table 5 Error rates on the Gun dataset for four different techniques

	Euclidian distance	DTW (ww = 100)	CDM	Bag-of-Patterns
4 Classes	0.40	0.20	0.05	0.00
2 Classes	0.01	0.00	0.00	0.00

4.2.3 UCR datasets and parameter training

Additionally, we conducted more comprehensive classification experiments using the datasets provided on the UCR Classification/Clustering Homepage (Keogh et al. 2006a). Since each dataset is divided into training set and testing set, we trained all the parameters from the training sets: n (sliding window size), w (number of segments), and α (alphabet size). For each dataset, we tried sliding window lengths that range between 15% and 36% of the time series length. For completeness, we tested w that range from 2 to $n/2$, doubling w each time (i.e. $w = 2, 4, 8 \ldots n/2$). The alphabet size, α, ranges from 3 to 10. We noticed that when the alphabet size is increased to a certain level, too many unique patterns are created, causing the classification quality to deteriorate. So we limited the size of pattern dictionary at 20,000. This helped speeding up the training by not considering alphabet sizes greater than the value that created this many patterns for any given n. Note that this threshold is optional – it is used merely to speed up the training. The parameters that result in the smallest errors on the training sets were used to classify the testing sets. The numerosity reduction option is turned off, since we observe that considering every subsequence results in slightly better accuracy for this experiment.

Once we determined the best parameters from training, we converted the test sets into the Bag-of-Patterns representation, and used 1-NN to classify the series. To be consistent with the results shown on the UCR Classification/Clustering website (Keogh et al. 2006a), we report the error rates instead of accuracy. Our results are shown as the last column in Table 6, along with the best parameters obtained from the training phase. For comparison, we also copied the error rates for the raw Euclidean Distance and DTW (with and without learning the warping window size) from the website. We also tried to use CDM to classify the data, but did not get very good results. However, the authors have made it clear that CDM is suitable for long time series (Keogh et al. 2004). Most time series in this data collection are too short for CDM to perform well.

For each dataset, the technique that yields the smallest error rate is highlighted. In case of a tie, all techniques with the best error rate are highlighted. The bottom row of the table shows the total number of "wins" for each technique: 9 for our approach, 3 for Euclidean distance on the raw data, 6 for DTW with warping window, and 8 for DTW with no warping window.

Overall, the Bag-of-Patterns and the constraint-less DTW have comparable classification accuracy. However, there is a significant difference between their time complexities. The linear complexity of the Bag-of-Patterns representation and its ability to capture structural information make it a promising representation for time series.

To get a better understanding on the type of datasets that the Bag-of-Patterns representation is more suitable for, we examined the OSU Leaf dataset, for which the Bag-of-Patterns approach achieved much better accuracy than the other methods. We examined class 1 and class 2, and randomly selected two time series from each of the two classes. The four time series are plotted in Fig. 17. The time series in the top plot are from class 2, and the time series in the bottom plot are from class 1. Looking at the data, it seems obvious why the shape-based methods fail: the data in the same class do not seem to resemble one another, or are out of phase. On the other hand, the Bag-of-Patterns approach is able to capture their structural differences. We observe that most time series that the Bag-of-Patterns approach does

Table 6 1-NN classification error rates. The error rates for the raw Euclidean distance and both versions of DTW are directly copied from the UCR Classification/Clustering website (Keogh et al. 2006a). The bottom of the table shows the total number of datasets for which each technique yields the smallest errors. Our approach has the highest number of wins compared to the other techniques

Name	Number of classes	Size of training set	Size of testing set	Time series Length	1-NN Euclidean Distance	1-NN Best Warping Window DTW (r)	1-NN DTW, no Warping Window	1-NN Bag of Patterns (n, w, a)
Synthetic control	6	300	300	60	0.12	0.017 (6)	0.007	0.037 (24, 4, 3)
Gun-point	2	50	150	150	0.087	0.087 (0)	0.093	0.027 (32, 4, 9)
CBF	3	30	900	128	0.148	0.004 (11)	0.003	0.013 (32, 4, 4)
Face (all)	14	560	1,690	131	0.286	0.192 (3)	0.192	0.219 (32, 8, 3)
OSU Leaf	6	200	242	427	0.483	0.384 (7)	0.409	0.256 (64, 4, 4)
Swedish Leaf	15	500	625	128	0.213	0.157 (2)	0.210	0.198 (40, 8, 4)
50Words	50	450	455	270	0.369	0.242 (6)	0.310	0.466 (80, 8, 3)
Trace	4	100	100	275	0.24	0.01 (3)	0.0	0 (48, 4, 4)
Two patterns	4	1,000	4,000	128	0.09	0.0015 (4)	0.0	0.129 (32, 4, 4)
Wafer	2	1,000	6,174	152	0.005	0.005 (1)	0.020	0.003 (32, 8, 5)
Face (four)	4	24	88	350	0.216	0.114 (2)	0.170	0.023 (64, 4, 4)
Lightning-2	2	60	61	637	0.246	0.131 (6)	0.131	0.164 (128, 8, 4)
Lightning-7	7	70	73	319	0.425	0.288 (5)	0.274	0.466 (64, 8, 4)
ECG	2	100	100	96	0.12	0.12 (0)	0.23	0.150 (32, 8, 4)
Adiac	37	390	391	176	0.389	0.391 (3)	0.396	0.432 (32, 8, 9)
Yoga	2	300	3,000	426	0.170	0.155 (2)	0.164	0.170 (80, 4, 9)
Fish	7	175	175	463	0.217	0.160 (4)	0.167	0.074 (128, 8, 7)
Beef	5	30	30	470	0.467	0.467 (0)	0.5	0.433 (80, 4, 2)
Coffee	2	28	28	286	0.25	0.179 (3)	0.179	0.036 (48, 4, 3)
Olive oil	4	30	30	570	0.133	0.167(1)	0.133	0.133 (160, 4, 6)
				# Wins	3	6	8	9

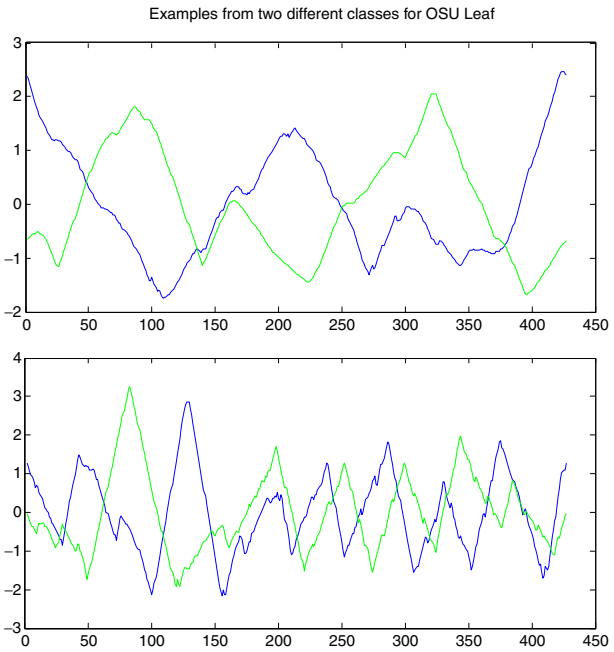

Fig. 17 (*Top*) Time series #3 and #15 from the OSU Training set. Both of them belong to class 2. (*Bottom*) Time series #133 and #187 from the OSU Training set. Both of them belong to class 1

well on are relatively long. In addition, at least four out of nine datasets for which the Bag-of-Patterns outperforms (or ties with) others are time series converted from shape data or video sequences. In other words, rotation or shift distortions might be present in the data, as Fig. 17 shows. In Section 5, we will investigate this further and demonstrate the utility of the Bag-of-Patterns representation on rotation-invariant matching. For now, we will contend ourselves by noting that the Bag-of-Patterns approach works better than shape-based approaches when data are out of phase, or when the ordering of the patterns is not critical (e.g. shape data, repetitive video sequences, and physiological measurements), whereas shape-based approaches work best when the ordering of the patterns is important (e.g. stock market data).

4.2.4 Distance measures for Bag-of-Patterns

In the next experiment, we investigate different distance measures that we can use on the bag of patterns representation. So far we have been using Euclidean distance, and it seems to work well. However, as cosine similarity is more commonly used for the bag of words representation for text data, we would like to see how cosine similarity compares to Euclidean distance for our representation. In addition, for text data, weighted term frequency is often recorded rather than raw term frequency. A well known term weighting scheme is called *tf-idf* (term frequency-inverse document frequency). With *tf-idf*, a term is weighted highly if it is frequent in relevant documents, but infrequent in the document collection as a whole. Terms that appear in virtually all documents would be weighted the lowest (Manning et al. 2008). Due to its popularity with text data, we would like to see whether this weighting scheme offers similar advantage for our data. We repeated the previous classification experiments on UCR data using the Bag-of-Patterns representation

Table 7 1-NN classification error rates for the Bag-of-Patterns representation using different distance measures and term weighting schemes. In our experiments, the Euclidean distance seems to outperform the commonly used cosine similarity, and raw frequency seems to outperform tf-idf

Name	Number of classes	Size of training set	Size of testing set	Time series Length	1-NN Bag of Patterns Euclidean (n,w,a)	1-NN Bag of Patterns Cosine Raw Freq (n,w,a)	1-NN Bag of Patterns Cosine tf-idf (n,w,a)
Synthetic control	6	300	300	60	0.037 (24, 4, 3)	0.047 (24,4,3)	0.060 (24,4,5)
Gun-point	2	50	150	150	0.027 (32,4,9)	0.002 (32,8,8)	0.040 (48,8,7)
CBF	3	30	900	128	0.013 (32,4,4)	0.013 (24,4,7)	0.013 (32,4,4)
Face (all)	14	560	1,690	131	0.219 (32,8,3)	0.230 (48,8,4)	0.219 (48,8,4)
OSU Leaf	6	200	242	427	0.256 (64,4,4)	0.331 (112,7,4)	0.236 (72,8,4)
Swedish Leaf	15	500	625	128	0.198 (40,8,4)	0.274 (40,8,6)	0.275 (40,8,6)
50Words	50	450	455	270	0.442 (96,8,3)	0.476 (112,8,3)	0.453 (104,8,3)
Trace	4	100	100	275	0 (48,4,4)	0 (48,4,4)	0.02 (48,4,4)
Two patterns	4	1,000	4,000	128	0.129 (32,4,4)	0.062 (32,4,3)	0.035 (48,4,5)
Wafer	2	1,000	6,174	152	0.003 (32,8,5)	0.106 (32,8,5)	0.015 (48,4,3)
Face (four)	4	24	88	350	0.023 (64,4,4)	0.046 (48,4,4)	0.011 (48,4,7)
Lightning-2	2	60	61	637	0.164 (128,8,4)	0.279 (216,4,6)	0.295 (216,8,6)
Lightning-7	7	70	73	319	0.466 (64,8,4)	0.466 (80,5,3)	0.356 (128,16,3)
ECG	2	100	100	96	0.150 (32,8,4)	0.140 (32,8,3)	0.140 (32,8,3)
Adiac	37	390	391	176	0.432 (32,8,9)	0.466 (32,8,9)	0.481 (40,8,8)
Yoga	2	300	3,000	426	0.170 (80,4,9)	0.433 (88,8,8)	0.202 (128,8,5)
Fish	7	175	175	463	0.074 (128,8,7)	0.086 (128,8,7)	0.114 (120,16,8)
Beef	5	30	30	470	0.433 (80,4,2)	0.400 (136,4,9)	0.400 (136,4,9)
Coffee	2	28	28	286	0.036 (48,4,3)	0.036 (48,4,3)	0.071 (48,4,3)
Olive oil	4	30	30	570	0.133 (160,4,6)	0.133 (104,16,3)	0.167 (120,4,3)
				# Wins	13	7	8

with the following variations: (1) raw frequency and Euclidean distance (this is reported in Table 6); (2) raw frequency and cosine similarity; and (3) tf-idf and cosine similarity (a typical combination for text data). The results are shown in Table 7. Similar to the previous experiment, we learned the best parameters using the training set. We copied the column "1-NN Bag of Patterns Euclidean" from the last column of Table 6 for easy comparison. Again, we record the number of "wins" for each technique at the bottom of the table. As the results demonstrate, Euclidean distance outperforms cosine similarity, and raw frequency outperforms weighted frequency. This is an interesting result, though not completely surprising, as the SAX words converted from time series might not have the same distribution as the terms in document data (i.e. Zipf distribution).

4.3 Discord/anomaly detection

A discord is defined as the data object that is the least similar to the rest of the dataset (Keogh et al. 2006b), i.e. it has the largest nearest neighbor distance. A discord can be seen as an anomaly in the data. In this section, we conducted discord/anomaly detection experiments, using our Bag-of-Patterns representation and comparing it with Euclidean distance on the raw data. The dataset we used is ECG2. We considered one class of ECG2 data at a time (recall that each class contains 50 ECG records), and manually inserted an anomaly by randomly choosing one other ECG record that belongs to a different class. For each class, we repeated this experiment 20 times, which resulted in a total of 100 runs. We then compared the accuracy, or the percentage of discords found, of our approach with the accuracy of Euclidean distance on the raw data. The accuracy results are shown in the last row of Table 4 above.

One of the reasons that our approach works so much better than the Euclidean distance is that the ECG data are not at all aligned, even for datasets in the same class. Another reason is that sometimes an ECG data might contain local anomalies within the data; such anomalies can easily throw off the distances computed using the shape-based approach.

We conclude this experiment by noting that the definition of discord can be easily extended to top k-discords. Applying k-discords discovery algorithm will allow us to find both global (i.e. data that do not belong in the cluster) and local (i.e. anomalies that occur within one time series) anomalies.

In next section, we demonstrate the utilities of the Bag-of-Patterns representation with a case study. More specifically, we demonstrate that the representation is rotation invariant, and that such invariance is particularly useful on certain types of data such as shapes.

5 Case study: rotation-invariant shape matching

With the rapid growth of computer storage technology, massive image or shape databases are created and stored for many disciplines such as medicine and science. The problem of efficiently searching and retrieving items that are similar to a given query from the database has been a focus of research in the past decade. One great challenge with shape mining and matching is that the approaches used must be

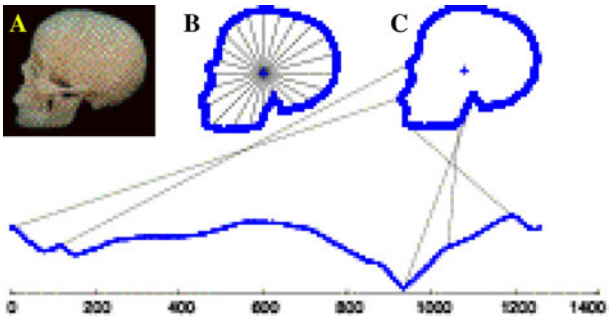

Fig. 18 A shape is converted to time series. The y-axis is the distance from the center of mass to the outline of the shape

invariant to different kinds of distortions such as rotation and scale (Keogh et al. 2006c). One way to represent shapes data is to convert them into sequences, or time series. Figure 18 shows an example of one such technique that converts a shape into a sequence of values (Keogh et al. 2006c). Recent studies have shown that such representations produce promising results for various data mining tasks (Keogh et al. 2006c; Wei et al. 2006)

While there are several variants of the conversion method, one advantage that is common to all of them is that the simple representation of time series makes it easier to handle many kinds of distortions, such as rotation. One approach to overcome rotation-related distortion is to create multiple copies of the original shape at different rotation angles, and exhaustively match the query shape against all the copies. This can be achieved easily if we convert the shape into a time series of length m. To generate all possible rotations, we simply shift the time series one point at a time (Keogh et al. 2006c), as illustrated by the matrix S below. The first time series $S_1 = [s_1, s_2, \ldots s_m]$ represents the original shape, and all the subsequent ones are its rotated versions (note the wrap-around at the end of the time series).

$$S = \begin{Bmatrix} s_1, s_2, s_3, \ldots, s_{m-2}, s_{m-1}, s_m \\ s_2, s_3, s_4, \ldots, s_{m-1}, s_m, s_1 \\ \vdots \\ \vdots \\ s_m, s_1, s_2, s_3, \ldots, s_{m-2}, s_{m-1} \end{Bmatrix} \tag{4}$$

This new dataset S represents the shape in all possible rotations. The rotation invariance is thus achieved by exhaustively searching through all the rotated versions of the shape. Clearly, this approach imposes a high computational cost, as the shape dataset is now m times larger (there are m copies for each shape). While there are techniques to speed up the search (Keogh et al. 2006c), we show that we can achieve rotation invariance *without* artificially generating different rotations. The intuition is to use the Bag-of-Patterns representation for the shape time series data. Since the Bag-of-Patterns representation consists of a combination of local patterns, the rotation information becomes irrelevant and not needed.

We used the five shape datasets from Table 6 in this experiment: Face (four), Face (all), Fish, OSU Leaf, and Swedish Leaf. To demonstrate the rotation-invariant property of our approach, we randomly rotated every object in the testing set. The classification error rates are reported in Table 8. The other methods that we compare with are the raw Euclidean distance, and DTW with 10% warping window size. For clarity, we also copied the results recorded on Table 6 for the un-rotated data. As

Table 8 1-NN classification error rates for randomly rotated shape data. The classification quality of our approach remains stable regardless of the rotations, whereas for the raw Euclidean distance and DTW, the quality worsens drastically

Name	Number of classes	Size of training set	Size of testing set (after random shift)	Time series Length	1-NN Euclidean Distance	1-NN DTW, 10% warping window	1-NN Bag of Patterns	BOP Error from Table 3
Face (four)	4	24	88	350	0.761 (0.719)	0.625	0.023 (0.023)	0.023
Face (all)	14	500	1690	131	0.893 (0.894)	0.812	0.223 (0.207)	0.219
Fish	7	175	175	463	0.823 (0.819)	0.857	0.040 (0.037)	0.074
OSU Leaf	6	200	242	427	0.562 (0.577)	0.496	0.260 (0.260)	0.256
Swedish Leaf	15	500	625	128	0.890 (0.889)	0.826	0.208 (0.207)	0.198

the results show, for the Bag-of-Patterns representation, the rotations did not make much difference in the classification accuracy. However, the classification quality drastically worsened for both the raw Euclidean distance and DTW. We repeated the experiments 4 times for the raw Euclidean distance and our approach, using different random rotations each time, and the results were very similar to what we obtained for the first random rotations. The averages from the four sets of random rotations are shown in parentheses.

Note if we had exhaustively searched all possible rotations, the results for the raw Euclidean distance and DTW would have been comparable to their respective results shown in Table 6 (see Keogh et al. 2006a for results on such experiments). However, our goal was to show that *without* increasing the sizes of the datasets by the artificial generation of different rotations, our approach did just as well as the distortion-free case. With this experiment, we conclude that our approach is indeed rotation-invariant without needing any manipulation of the data.

6 Conclusion

Most existing work on time series similarity search focuses on finding shaped-based similarity. While these shape-based approaches work reasonably well for short time series data, the accuracy typically degrades if the sequences are long. For long time sequences, it is more appropriate to measure the similarity by looking at their higher-level structures, rather than point-to-point, local comparisons. The need for structure-based representation is similar to that for textual data: if we are to compare two documents, it's more meaningful to use a higher-level representation instead of comparing strings using edit distance.

In this work, we proposed a structural similarity measure using a histogram-based representation. Similar to the bag-of-words representation for textual data, our approach counts the frequency of occurrences of each pattern in the time series. We then compare the frequencies (or the histograms) of patterns in the time series to obtain a similarity measure.

Our experimental results show that our approach is superior to existing leading approaches in the tasks of clustering, classification, and discord discovery. Furthermore, our approach has several advantages over existing structure-based similarity measures. Specifically, our approach considers local structures as well as global structure, by using subsequences to build our final representation. Our representation allows users to understand the pattern distribution by examining the histograms. Furthermore, our representation is suitable for streaming data, since the frequency vectors are built incrementally. We demonstrate that the representation also allows rotation-invariant matching without having to manipulate the data.

We would like to note that since our approach determines similarity based on structures of the data, the input sequences should be reasonably long, or long enough such that the structures (or lack of structures) can be meaningfully captured and summarized. For short time series, off-the-shelf distance measures such as Dynamic Time Warping, Euclidean distance or dimensionality reduction techniques work reasonably well. This coincides with the observations made by Keogh et al. (2004). Nevertheless, our representation does well on some of the short time series.

For future work, we would like to extend the representation for multivariate time series data. In addition, instead of extracting fixed-length subsequences to

form the pattern vocabulary, we would like to explore the potential to extract more meaningful, variable-length patterns.

7 Integrity of research

All experiments conducted in this paper comply with the current laws of the United States.

The authors declare that they have no conflict of interest.

References

Agrawal, R., Faloutsos, C., & Swami, A. (1993). Efficient similarity search in sequence databases. In *Proceedings of the 4th int'l conference on foundations of data organization and algorithms* (pp. 69–84). Chicago, IL.

Bradley, P., Fayyad, U., & Reina, C. (1998). Scaling clustering algorithms to large databases. In *Proceedings of the 4th int'l conference on knowledge discovery and data mining* (pp. 9–15). New York, NY.

Chan, K., & Fu, A. W. (1999). Efficient time series matching by wavelets. In *Proceedings of the 15th IEEE int'l conference on data engineering* (pp. 126–133). Sydney, Australia.

Chen, L., & Ng, R. (2004). On the marriage of Lp-norms and edit distance. In *Proceedings of the thirtieth international conference on very large data bases* (Vol. 30, pp. 792–803).

Crochemore, M., Czumaj, A., Gasjeniec, L., Jarominek, S., Lecroq, T., Plandowski, W., et al. (1994). Speeding up two string-matching algorithms. *Algorithmica, 12*, 247–267.

Deng, K., Moore, A., & Nechyba, M. (1997). Learning to recognize time series: Combining ARMA models with memory-based learning. *IEEE International Symposium on Computational Intelligence in Robotics and Automation, 1*, 246–250.

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., & Keogh, E. (2008). Querying and mining of time series data: Experimental comparison of representations and distance measures. *Proceedings of VLDB Endowment, 1*(2), 1542–1552.

Faloutsos, C., Ranganathan, M., & Manolopulos, Y. (1994). Fast subsequence matching in time-series databases. *SIGMOD Record, 23*, 419–429.

Gavrilov, M., Anguelov, D., Indyk, P., & Motwahl, R. (2000). Mining the stock market: Which measure is best? In *Proceeding of the 6th ACM SIGKDD*.

Ge, X., & Smyth, P. (2000). Deformable Markov model templates for time-series pattern matching. In *Proceedings of the 6th ACM SIGKDD* (pp. 81–90). Boston, MA.

Geurts, P. (2001). Pattern extraction for time series classification. In *Proceedings of the 5th European conference on principles of data mining and knowledge discovery* (pp. 115–127). Freiburg, Germany.

Goldberger, A. L., Amaral, L., Glass, L, Hausdorff, J. M., Ivanov, P. Ch., Mark, R. G., et al. (1997). PhysioBank, PhysioToolkit, and PhysioNet: Circulation 101(23):e215–e220. *Discovery, 1*(3).

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika, 2*, 241–254.

Keogh, E. (2002). Exact indexing of dynamic time warping. In *Proceedings of the 28th international conference on very large data bases*. Hong Kong, China.

Keogh, E. (2004). Tutorial in SIGKDD. In *Data mining and machine learning in time series databases*.

Keogh, E., & Kasetty, S. (2002). On the need for time series data mining benchmarks: A survey and empirical demonstration. In *Proceedings of the 8th ACM SIGKDD international conference on knowledge discovery* (pp. 102–111). Edmonton, Alberta, Canada.

Keogh, E., Chakrabarti, K., & Pazzani, M. (2001) Locally adaptive dimensionality reduction for indexing large time series databases. In *Proceedings of ACM SIGMOD conference on management of data* (pp. 151–162). Santa Barbara.

Keogh, E., Lonardi, S., & Ratanamahatana, C. A. (2004). Towards parameter-free data mining. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining*. Seattle, WA, USA.

Keogh, E., Xi, X., Wei, L., & Ratanamahatana, C. (2006a). The UCR time series classification/clustering homepage. http://www.cs.ucr.edu/~eamonn/time_series_data. Accessed 12 July 2011.

Keogh, E., Lin, J., & Fu, A. (2006b). *Finding the most unusual time series subsequence: Algorithms and applications. Knowledge and Information Systems (KAIS)*. Springer-Verlag.
Keogh, E., Wei, L., Xi, X., Lee, S., & Vlachos, M. (2006c). LB_Keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures. In *Proceedings of the 32nd international conference on very large data bases*.
Ko, M. K., West, G., Venkatesh, S., & Kumar, M. (2005) Online context recognition in multisensor systems using dynamic time warping. In *Intelligent sensors, sensor networks and information processing conference* (pp. 283–288).
Kriegel, H.-P., Kroger, P., Pryakhin, A., Renz, M., & Zherdin, A. (2008). Approximate clustering of time series using compact model-based descriptions. In J. R. Haritsa, R. Kotagiri, & V. Pudi (Eds.), *Proceedings of the 13th international conference on database systems for advanced applications (DASFAA'08)* (pp. 364–379). Berlin, Heidelberg: Springer-Verlag.
Li, M., & Vitanyi, P. (1997). *An introduction to Kolmogorov complexity and its applications*, 2nd Edn. Springer Verlag.
Lin, J., & Li, Y. (2009). Finding structural similarity in time series data using Bag-of-Patterns representation. In M. Winslett (Ed.), *Proceedings of the 21st international conference on scientific and statistical database management (SSDBM 2009)* (pp. 461–477). Berlin, Heidelberg: Springer-Verlag.
Lin, J., Keogh, E., Li, W., & Lonardi, S. (2007). Experiencing SAX: A novel symbolic representation of time series. *Data Mining and Knowledge Discovery, 15*(2), 107–144.
Lin, J., Vlachos, M., Keogh, E., & Gunopulos, D. (2004). Iterative incremental clustering of time series. In *IX conference on extending database technology (EDBT)*.
Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.
Marteau, P.-F. (2009). Time warp edit distance with stiffness adjustment for time series matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 31*(2), 306–318.
McQueen, J. (1967). Some methods for classification and analysis of multivariate observation. In L. Le Cam, & J. Neyman (Eds.), *Proceedings of the 5th Berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 281–297). Berkeley, CA.
Mueen, A., Keogh, E., & Young, N. (2011). Logical-shapelets: An expressive primitive for time series classification. In *Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining (KDD '11)*. San Diego, CA.
Nanopoulos, A., Alcock, R., & Manolopoulos, Y. (2001). Feature-based classification of time-series data. In N. Mastorakis, & S. D. Nikolopoulos (Eds.), *Information processing and technology* (pp. 49–61). Commack, NY: Nova Science Publishers.
Olszewski, R. (2001). Generalized feature extraction for structural pattern recognition in time-series data. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.
Radovanovic, M., Nanopoulos, A., & Ivanovic, M. (2010). *Time-series classification in many intrinsic dimensions* (pp. 677–688). SDM.
Ratanamahatana, C. A., & Keogh, E. (2004). Making time-series classification more accurate using learned constraints. In *Proceedings of SIAM international conference on data mining*. Lake Buena Vista, Florida.
Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM, 19*, 613–620.
Sart, D., Mueen, A., Najjar, W., Keogh, E., & Niennattrakul, V. (2010). Accelerating dynamic time warping subsequence search with GPUs and FPGAs. In *Proceedings of the 2010 IEEE international conference on data mining (ICDM'10)* (pp. 1001–1006). Washington, DC, USA.
Wang, X., Smith, K., & Hyndman, R. (2006). Characteristic-based clustering for time series data. *Data Mining and Knowledge Discovery, 13*(3), 335–364.
Wei, L., & Keogh, E. (2006). Semi-supervised time series classification. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 748–753). New York, NY, U.S.A.: ACM.
Wei, L., Keogh, E., & Xi, X. (2006) SAXually explicit images: finding unusual shapes. In *Proceedings of the IEEE international conference on data mining*. Hong Kong.
Vlachos, M., Gunopoulos, D., & Kollios, G. (2002). Discovering similar multidimensional trajectories. In *Proceedings of the 18th International Conference on Data Engineering*.
Xing, Z., Pei, J., Yu, P., & Wang, K. (2011). Extracting interpretable features for early classification on time series. In *Proceedings of SDM, 2011*.
Ye, L., & Keogh, E. (2009). Time series shapelets: A new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining (KDD'09)*. New York, NY.