Time series feature extraction for data mining using DWT and DFT

Fabian Mörchen [*]

November 5, 2003

Abstract

A new method of dimensionality reduction for time series data mining is proposed. Each time series is compressed with wavelet or Fourier decomposition. Instead of using only the first coefficients, a new method of choosing the best coefficients for a set of time series is presented. A criterion function is evaluated using all values of a coefficient position to determine a good set of coefficients. The optimal criterion function with respect to energy preservation is given. For many real life data sets much more energy can be preserved, which is advantageous for data mining tasks. All time series to be mined, or at least a representative subset, need to be available a priori.

Keywords: time series, data mining, feature extraction, wavelet, Haar, Fourier, clustering, classification, rule generation

1 Introduction

A big problem when mining in time series data is the high dimensionality. It is usually not enough to look at each point in time sequentially, rather one has to deal with sliding windows of a possibly multi-dimensional time series. By enlargening the context window, this quickly produces very high dimensional vectors, introducing the *curse of dimensionality* and causing problems with distance metrics [5], [1]. The large amount of data also heavily

[*] Fabian Mörchen, Data Bionics, Philipps-University Marburg, Hans-Meerwein-Strasse, 35032 Marburg, Germany (email: fabian@mybytes.de)

slows down the algorithms or even makes them useless. For example, the performance of many time series indexing methods using spatial indexing techniques deteriorates with large dimensionality [3] such that a sequential scan of the database will be faster.

Luckily, consecutive values of a time series are usually not independent but highly correlated, thus there is a lot of redundancy. Feature extraction should be applied to compress the time series, keeping only the important information while discarding noise and removing correlations. If the right features are extracted, data mining algorithms will not only be speeded up, they will produce better results than on the original data. Especially clustering algorithms will profit from the data cleaning. A reduction down to a few features also increases the interpretability of the results. When using rule generation algorithms, fewer dimensions will produce more understandable rules. Having humans understand, what data mining algorithms find, is the ultimate goal of knowledge discovery, after all.

Popular feature extraction techniques for time series include the Discrete Wavelet Transform (DWT) and the Discrete Fourier Transform (DFT). The signal is projected into the frequency domain (DFT) or a tiling of the time-frequency plane (DWT). By keeping only the first few coefficients as features, each time series is represented by a rough sketch, because these coefficients correspond to the low frequencies of the signal. But using the first coefficients is not the best way to compress time series data. Using the largest coefficients instead, will preserve the optimal amount of energy present in the original signal [31]. When dealing with several times series, however, this introduces higher storage demands and/or bookkeeping overhead for the distance computations. These disadvantages made Wu *et. al.* rule out this technique, at least for the task of time series indexing [31].

We propose a new method of choosing the *same* subset of the coefficients for all time series, to achieve the same dimensionality reduction as the traditional method but keeping the distance computation simple. Compared to the optimal method of keeping the largest coefficients per time series individually, the extracted features are more comparable among the time series, any results will thus be more interpretable. The coefficients are selected using a criterion function over all time series. In order to deal with abstracted versions of the time series, that are as similar as possible to the original data, energy preservation will be optimized. This method is only applicable if all time series, or at least a representative subset, is available beforehand. This is not a restriction for many data mining techniques. We will discuss the implications for data mining in the context of clustering, classification, and indexing.

The rest of this paper is structured as follows. In Section 2 we briefly review related work. Section 3 gives some definitions and theoretical background. Section 4 describes the proposed feature selection in detail. After a brief description of the data in Section 5 the experimental results are presented in Section 6. The results and implications are discussed in Section 7 before the final conclusions are drawn in Section 8.

2 Related Work

A lot of research has been done on time series similarity measures and feature extraction with DFT and DWT in the context of time series indexing. Usually the problem posed is the search for patterns in a time series database similar to a given pattern (*query by example*). The pioneering work by Agrawal *et. al.* [3] proposed to use the first few DFT coefficients indexed in a R^*-Tree [4]. Since the Euclidean distance on the DFT coefficients lower bounds the Euclidean distance on the original time series, there are no false dismissals. Any false hits are removed in a post-processing step using the original data.

Many follow up papers extended this approach, e.g. to handle scaling and gaps [2], subsequence matching using minimum bounding rectangles [9], [21], formalizing query constraints and incorporating them into the indexing procedure [12], [24], using the last k DFT coefficients with the conjugate property of the DFT [25], or using Haar DWT instead of DFT [6]. The competition between DFT and Haar DWT was resolved by Wu *et. al.* [31], concluding that they are comparable in energy preservation, but that DWT is faster to calculate and offers a multi-resolution decomposition. In addition there are approaches to combine DWT with time warping [7].

Popivanov *et. al.* [22] pursued the suggestion by Chan [6], that wavelet bases other than Haar, might be better for certain data sets. They showed that the contraction property needed for indexing holds for all (bi-)orthogonal wavelet transforms, making a large family of transforms available for the feature extraction. In particular the Daubechies wavelet family [8] was shown to have a better energy preservation on several real life and synthetic datasets.

One possibility to search for groups in a set of time series is clustering. The approximate distance functions from the indexing literature and other sophisticated distance functions have been combined with clustering algorithms, for example k-Means clustering [13], [15] (with ARIMA based distance), [20] (with DWT), hierarchical clustering [30] (with Euclidean distance), [27] (with dynamic time warping distance), [13], or SOM clustering [10] (with perceptually important point distance). Keogh *et. al.* [17] tested many sophisticated distance functions on two benchmark datasets and found them not to be better than Euclidean distance. Note, that when clustering subsequences, one should not use overlapping sliding windows, because it has been shown to produce meaningless results [18].

Once groups are known, it is interesting to find descriptions of the groups to extract knowledge or classify new data. Classification of time series is done in [26] by generating rules with inductive logic programming and in [11] by regression trees and decision trees. Kadous [14] also uses decision trees and emphasizes comprehensibility of the results.

3 Background

3.1 Time-Frequency Transforms

Let $(f(1), ..., f(n))$ be a uniform sample of a real signal $f(t)$. Both the Fourier Transform (FT) and the Wavelet Transform (WT) express the signal as coefficients in a function space spanned by a set of basis functions. The basis of the FT contains only the complex exponential function, representing sinusoid functions in the real domain. The basis of the WT consists of infinitely many scaled and shifted versions of a mother wavelet function, often with compact support. For both transforms a discrete version exists and will be reviewed here briefly.

The *Discrete Fourier Transform* (DFT) is the projection of a signal from the time domain into the frequency domain by

$$c_f = \frac{1}{\sqrt{n}} \sum_{t=1}^{n} f(t) \exp\left(\frac{-2\pi i f t}{n}\right) \qquad (1)$$

where $f = 1, ..., n$ and $i = \sqrt{-1}$. The c_f are complex numbers and represent the amplitudes and shifts of a decomposition of the signal into sinusoid functions. For real signals, c_i is the complex conjugate of c_{n-i+1} $(i = 2, \ldots, \frac{n}{2})$. The Fourier transform measures global frequencies and the signal is assumed to be periodic. The latter assumption can cause poor approximation at the borders of a time series. There is a fast algorithm, the *Fast Fourier Transform* (FFT), with time complexity $O(n \log n)$.

The *Discrete Wavelet Transform* (DWT) measures frequency at different time resolutions and locations. The signal is projected into the time-frequency plane. The basis functions are

$$\Psi_{j,k}(t) = 2^{\frac{j}{2}} \Psi \left(2^j t - k \right) \tag{2}$$

where Ψ is the mother wavelet function. Any square integrable real function $f(t)$ can be represented in terms of this bases as

$$f(t) = \sum_{j,k} c_{j,k} \Psi_{j,k}(t) \tag{3}$$

and the $c_{j,k} = \langle \Psi_{j,k}(t), f(t) \rangle$ are the coefficients of the DWT. There is a fast algorithm, working with a dyadic grid of resolutions with time complexity $O(n)$ [19]. A simple and commonly used wavelet is the Haar wavelet with the mother function

$$\Psi_{Haar}(t) = \begin{cases} 1, & \text{if } 0 < t < 0.5 \\ -1, & \text{if } 0.5 < t < 1 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

3.2 Definitions

For a set of l time series let $C = (c_{i,j})$ $i = 1, ..., l$ $j = 1, ..., m$ be the $l \times m$ matrix of the coefficients obtained using either the DFT or the DWT. For DFT and the Haar DWT $m = n$, but for some wavelet bases $m > n$ due to additional coefficients needed at the boundaries. Let c_j be the j-th column of C. Let $I : \mathbb{R}^l \mapsto \mathbb{R}$ be a function measuring the importance of the coefficient position j based on all l values from this coefficient. Let $J_k(I, C)$ be a function choosing a subset of $M = \{1, ..., m\}$ of size k based on the aggregate values calculated with I on all columns of C. There are two sensible choices for J_k, choosing either the columns with the k largest or the k smallest values of I, depending on the semantic of I:

$$J_k^1(I, A) = \{J | I(c_j) \geq I(c_{j'}) \; \forall j \in J, j' \in M \setminus J\} \tag{5}$$

$$J_k^2(I, A) = \{J | I(c_j) \leq I(c_{j'}) \; \forall j \in J, j' \in M \setminus J\} \tag{6}$$

where $J \subset M$ with $|J| = k$.

The energy of a sampled signal is defined as

$$E(f(t)) = \sum_{t=1}^{n} f(t)^2 \tag{7}$$

For for orthogonal transforms, such as the DFT and the DWT (for certain wavelet bases) this energy is preserved, thus

$$E(f(t)) = \sum_{j=1}^{m} a_j c_j^2 \tag{8}$$

with appropriate scaling coefficients a_j for the coefficients c_j obtained from the corresponding signal $f(t)$.

4 Feature Extraction

Using the point wise Euclidean distance between two time series for data mining is problematic. The high dimensionality, produced by using each and every time point, makes the concept of a nearest neighbor meaningless [5], unless using very short time series.

Any results of data mining, for example generated rules describing some classes, will also be highly questionable. While automated algorithms could still process conditions on many single time points, they will hardly be understandable by humans. But extracting knowledge that is interpretable by a domain expert is the ultimate goal of knowledge discovery. Even if there were rules based on only a few time points, it would be unclear why these particular points have been selected. Any time warping effects in new, unclassified data would make the rules useless, because some phenomena might not occur at these exact time locations anymore.

Clustering algorithms rely on a meaningful distance function to group data vectors that are close to each other and distinguish them from others that are far away. But in high dimensional spaces the contrast between the nearest and the farthest neighbor gets increasingly smaller [5], making it impossible to find meaningful groups.

Finally, many data mining algorithms, will also be heavily slowed down. The number of dimensions usually affects the time complexity at least linearly. Sometimes the slowdown reaches a point where the method becomes useless. Spatial indexing techniques commonly used in time series indexing often suffer so baldy from many dimensions that a sequential scan of the database will be faster than using the index.

But using all time points isn't really necessary, because time series usually show a lot of redundancy and noise that should be removed. A solution to all the problems mentioned above is to reduce the time series to carefully selected features, that are influenced by the whole time series or parts of it. From an information theoretic point of view this reduces the redundancy, from a signal processing point of view this will remove noise, and from a knowledge discovery point of view it offers a higher semantic power for possible results. Thus feature extraction is commonly used to reduce the dimensionality and make distance calculations meaningful and feasible for data mining algorithms. Algorithms run faster and produce results that are more understandable and based on meaningful distance relations.

A well established feature extraction technique using DFT and DWT for time series is to use only the first k coefficients, discarding the rest. This corresponds to saving only a rough sketch of the time series, because the first coefficients represent the low frequencies of the signal. For the DFT each features is influenced by all time points, because the coefficients describe global frequencies. For the DWT the coefficients are influenced by sub-series of different sizes, offering a multi-resolution decomposition.

A much better technique is to use the k largest coefficients of each time series, because they preserve the optimal amount of energy per time series [31]. Better energy preservation means, that a prototype, reconstructed using the extracted features, will be more similar to the original signal with respect to the Euclidean distance. Thus it also will preserve the Euclidean distance between two time series better, an important property for data mining. Keeping the largest instead of the first coefficients is common practice when compressing a single signal, but it causes problems when dealing with several times series. It introduces higher storage demands and/or bookkeeping overhead for the distance computations. These disadvantages led Wu *et. al.* to conclude that this technique is not suitable for time series indexing [31].

One approach to keep the optimal amount of energy would be to save the positions of the selected coefficients along with the values for each times series. This produces only a moderate memory overhead (because the positions are natural numbers) but the distance computation has to keep track of the positions to compare only the corresponding positions (and assuming a coefficient not present in one of the time series to be zero). Calculating the distance will thus be much more expensive. The set of features available to rule generation algorithms will be the union of the best k coefficient positions over all time series, thus generally be significantly larger than k. This is likely to produce more complex, less understandable rules.

Simply keeping the union of the best k coefficient positions (and optionally setting the coefficients not belonging the the largest k in each times series to zero) removes the need for bookkeeping, but will result in a significant memory overhead and it still undermines the aim of producing a small number of features. The distance computation will obviously also be slowed down as well.

One could also keep a different number of coefficients for each time series, either determined by a threshold on the energy that should to be preserved per time series or by simply choosing the kl largest coefficients of C to maximize the total energy preserved. This has the same disadvantages as the above method of choosing the k largest per series.

We propose to use the *same* subset of the coefficients of size k, chosen by an aggregate function, measuring the importance of each coefficient position. The criterion function is evaluated with the values at a coefficient position from all times series. This way we achieve the same dimensionality reduction down to k values per series and keep the distance computation simple. The restriction to the same subset increases comparability of two time series and thus the interpretability of any results.

Compared to choosing the first coefficients, it also offers more semantic power in the sense, that more complex shapes can be represented by a feature vector. By using arbitrary coefficient positions not only low frequencies but any frequency can be included if it is of importance in the data set. The locality of the DWT even makes it possible to zoom into important locations in the time series, representing it with many features and approximating other less important parts with fewer values.

Note that this technique is data dependent, in the sense that it can only be used if all time series, or at least a representative subset, is available a priori, but this is not a restriction for the task of data mining. A desirable optimization goal for the criterion function is clearly energy preservation, because the more energy is preserved, the more similar the series reconstructed from the features are to the original data.

Theorem 1 *When choosing k columns of C the function $J_k^1(mean(c_j^2), C)$ is optimal in energy preservation.*

Proof: We want to optimize the energy present in the coefficients of the columns indexed by $J \subset M$:

$$E = \sum_{i=1}^{l} \sum_{j \in J} c_{i,j}^2 \tag{9}$$

Interchanging the sums and factorizing we get a term proportional to the sum of the mean power per coefficient position

$$E = \sum_{j \in J} \sum_{i=1}^{l} c_{i,j}^2 = \sum_{j \in J} (l \frac{1}{l} \sum_{i=1}^{l} c_{i,j}^2) = l \sum_{j \in J} mean(c_j^2) \propto \sum_{j \in J} mean(c_j^2) \tag{10}$$

where c_j^2 is the element wise square of c_j. Thus choosing $I(c_j) = mean(c_j^2)$ and using J_k^1 will give the best energy preservation with k coefficients possible, under the restriction of choosing the same subset for all time series.□

How much energy preservation is actually gained, when using this method compared to keeping the first coefficients, and how close it is to the optimal method of keeping the kl largest coefficients over all series, will be tested in Section 6.1 on several real life and synthetic data sets.

Note that one immediate advantage of our method for wavelet coefficients is the fact, that any number of coefficients can be used. When choosing the first coefficients, it does not make sense to split a resolution level. Using the Haar wavelet, for example, the first coefficient represents the average value of the time series. The first detail level contains one coefficient representing the difference between the averages of the first and second half of the signal. The succeeding detail levels always add double as many coefficients as the previous one. When transforming a time series of length 32, for example, the coefficients are partitioned into blocks of the following sizes $1, 1, 2, 4, 8, 16$ corresponding to increasing resolution levels. A sensible number of coefficients to use can only be $1, 2, 4, 8, 16, 32$ while the proposed selection technique could use any number between 1 and 32. It is very common to normalize all time series to have a mean value of 0. Then the first coefficient is not needed and the number of coefficients can be $1, 3, 7, 15, 31$. This technique was used in the experiments in Section 6.1. For Daubechies wavelets of higher order the increasing filter length causes even coarser steps. Note, that this method of not splitting levels was not followed in [22], instead a dimensionality of 9 was used for most experiments.

5 Data

How much more energy is preserved by the new method will depend on the data, namely on the important frequency ranges and positions in the signal. We used the largest datasets from the UCR Time Series Data Mining Archive (TSDMA) [16] to get sets of time series with high dimensionality and to cover a variety of applications.

For each time series the data was transformed into samples of length 1024. Each dimension of each time series that had at least 1024 values was segmented with a non-overlapping window, remaining values at the end were discarded. For time series that had less than 1024 time points, the last values were mirrored. Time series with less than 10 samples extracted by this procedure were discarded. In Table 1 all data sets used are listed with their original size and the number of samples of length 1024 that were created from them.

Additional tests were made in order to evaluate the proposed method for clustering and classification with rules. The quality measure for clustering was the classification error of the resulting partition, while for the classification rules the main concern was the complexity of the rule set. To measure the classification error, datasets with a prior classification are needed. Unfortunately, classified sets of time series are hard to find. We used the Heterogeneous dataset from [20]. It is constructed out of 10 different datasets from the UCR TSDMA. Each class contains examples from one of 10 original datasets deteriorated by noise and local warping. We used 100 samples per class for clustering and 80 samples per class for the rule generation, split in half for training and testing.

6 Experiment

6.1 Energy Preservation

Each time series data set was compressed using four methods. The first one, called first_k , is the traditional method of keeping the first k coefficients, that

Name	**Size**	**Samples**	**Problem Domain**
buoy_sensor	13991×4	52	Buoy sensor data
cstr	7500×3	21	Continuous stirred tank reactor
eeg	512×22	21	EEG data
ERP_data	6400×31	186	ERP data
evaporator	6305×6	36	Industrial evaporator process
foetal_ecg	2500×9	16	Cutaneous potential recordings
koski_ecg	144002×1	140	ECG
network	18000×1	17	Network packet round trip
pgt50_alpha	990×18	18	Gene expression
pgt50_cdc15	990×24	24	Gene expression
power_data	35040×1	34	Power demand
random_walk	65536×1	64	synthetic
spot_exrates	2567×12	24	Currency spot prices
sp500	17610×2	17	S&P500 stock index
steamgen	9600×4	36	Power plant steam generator
synthetic_ctrl	600×60	60	synthetic
tickwise	279113×1	272	Exchange rate

Table 1: Time series used for energy preservation tests

is selecting the coefficients level wise starting with the coarsest resolution level. The second one is the proposed method of using the proposed energy criterion, called $best_k$. This will always be better than the first method. The question is whether the additional effort of calculating the energy pays off in significantly better energy preservation. The third method, called $each_k$, is the locally optimal procedure of keeping the k largest coefficients for each time series individually and the last method, called $adap_k$, is the globally optimal procedure of keeping the kl largest coefficients over all time series. The latter two were used in spite of the disadvantages discussed above, to see how close the energy preservation of the first two methods is to the optimal amount of energy that can be preserved with k coefficients per series or a total of kl coefficients, respectively. Note that only the $first_k$ method is restricted to the values of k used here. All other methods can be used with any $k = 1, ..., m$, but for better comparison the same steps were used.

To evaluate the quality, the energy preservation of each method is measured as the percentage of the total energy present in the set of the original signals. The *gain* is measured as the difference of the percentages of the two competing methods. The difference of the energy preserved by the optimal method and the proposed method is called *left*. All percentages are given with respect to the original energy, not with respect to the $first_k$ method. Measuring the gain in percent with respect to the traditional method would often give much higher values, but this would be over emphasizing the results and hinder the comparison to the optimal $adap_k$ method. Note, that the chosen quality measure of energy preservation directly corresponds to measuring reconstruction error and thus to preservation of the distances between the time series, so these performance measures need not be evaluated separately.

The coefficients were calculated with the DFT and the DWT using the the Haar and some higher order Daubechies bases. Note that we do not compare the energy preservation of a signal by the these time-frequency transforms directly because it has been done in [31] and [22] and is beyond the scope of this paper. In general DFT will work better for signals with strong regular global patterns, while the multi-resolution property of the DWT makes it suitable for signals with locally occuring patterns.

In Table 2 you can see the results for the *foetal_ecg* example compressed with the Haar DWT. For $k \in \{3, 7, 15, 31, 63, 127, 255\}$ [1] between 9% and 48% more energy of the original signal is preserved by the proposed method (see

[1] the first three values of k represent a dimensionality typically used for indexing

also Figure 1). Looking at an example of these time series in Figure 2 shows why: There are clearly dominant high frequencies that hold a lot of energy. When choosing coefficients level wise, the high frequencies will be selected rather late, only after many low, rather unimportant frequencies have been selected. The selection by energy successfully re-orders the coefficients in a way that these important high frequencies are chosen earlier.

Even though much more energy is preserved by the proposed method, there is still room for improvement. For $k \in \{7, 15, 31\}$ the optimal method $adap_k$ can preserve at least another 20% of the original energy. Whether this further gain is worth the effort of the more complicated distance calculation while loosing interpretability at the same time is application dependent.

Note, that the difference between the $adap_k$ method and the $each_k$ method is really small. For this dataset the adaptive method is not really needed, keeping k coefficients per time series is at most 0.7% worse. This effect was observed on all examples and might be due to the rather uniform nature of each dataset.

k	$adap_k$	$each_k$	$best_k$	$first_k$	gain	left
1	11.6	10.9	4.6	1.9	**2.7**	7.0
3	25.3	24.9	12.6	3.2	**9.5**	12.7
7	43.7	43.1	22.5	4.6	**17.9**	21.2
15	65.3	64.7	37.4	6.5	**30.9**	27.8
31	82.1	82.0	59.4	12.3	**47.1**	22.8
63	92.2	91.9	78.9	30.8	**48.1**	13.3
127	97.4	97.2	91.2	61.9	**29.3**	6.1
255	99.2	99.2	97.3	85.5	**11.8**	1.9
511	99.9	99.8	99.4	96.5	**2.9**	0.5
1023	100.0	100.0	100.0	100.0	**0.0**	0.0

Table 2: Energy preservation with Haar DWT for foetal_ecg

The gain achieved by the $best_k$ method is not always as large as for the *foetal_ecg* data. A particularily bad example is the energy preservation of the S&P 500 stock index (*sp500*) shown in Figure 3. Here almost no improvement with the selection by energy is achieved over the traditional method. The curves for $first_k$ and $best_k$ are almost identical. They are also very close to the optimal $adap_k$, which means, that the most dominant frequencies in this

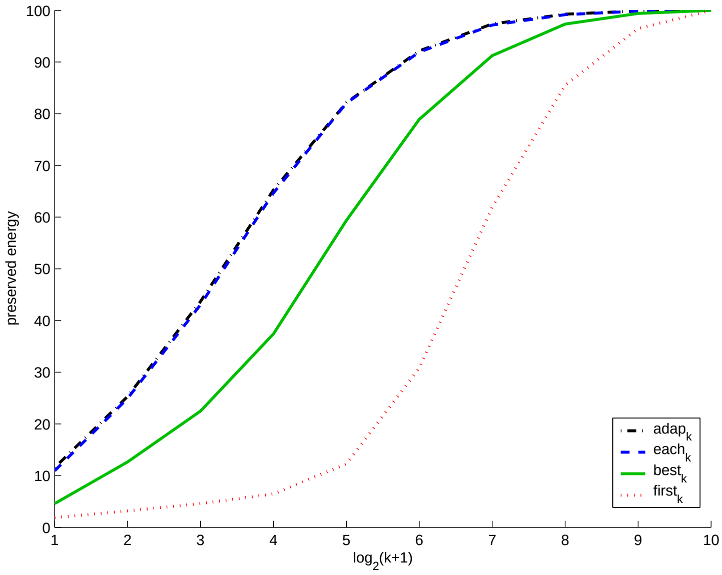

Figure 1: Energy preservation with Haar DWT for foetal_ecg

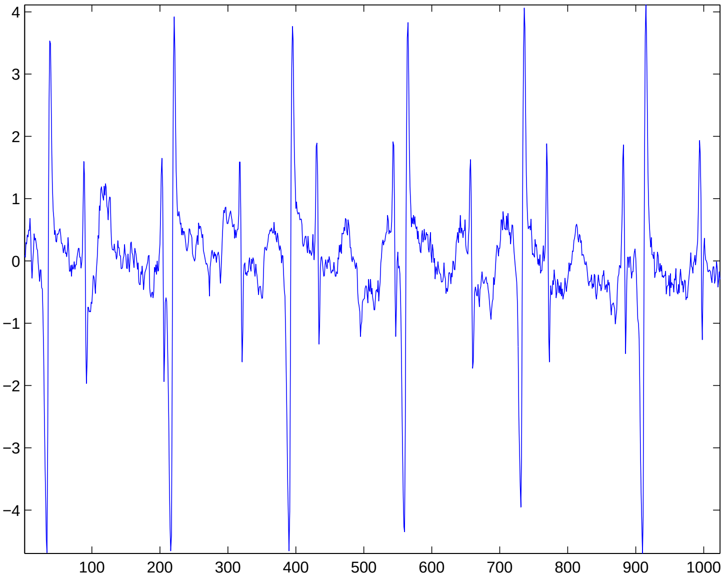

Figure 2: Example time series of foetal_ecg

signal are in fact the low frequencies, as one can see in Figure 4. Thus here is simply not much to improve.

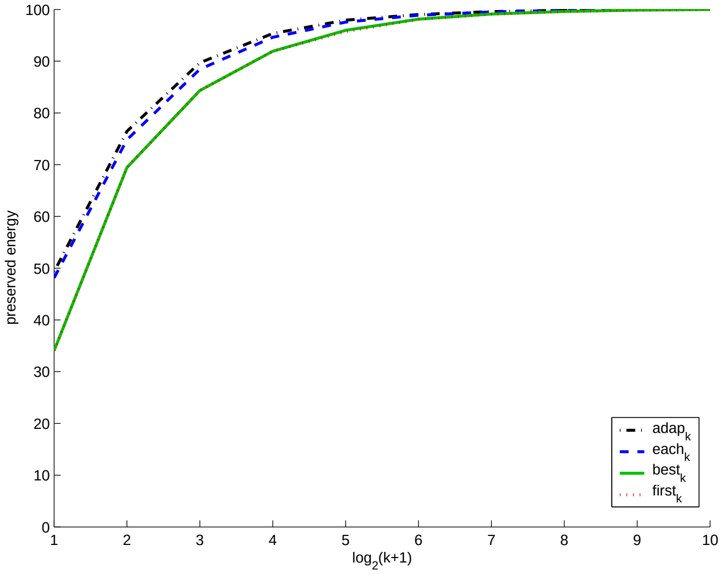

Figure 3: Energy preservation with Haar DWT for sp500

The results for all time series data sets and the Haar DWT are listed in Table 3. The percentage differences listed correspond to the bold column of Table 2, that is the additional energy preserved by $best_k$ over $first_k$. The column with 1023 coefficients was omitted because using all coefficients always preserves all the energy.

Medical data seems to profit from the new method quite well. Apart from the excellent improvement of the *foetal_ecg* data set, the *ERP_data* example has a gain of around 12% for the coefficient numbers 7 through 31 and the *eeg* example inhibits a gain of around 6% from 3 up to 31 coefficients. But there are more positive examples from other applications: the *synthetic_ctrl* data set shows good values throughout the range of coefficients tested and the *network* and the *pgt50* examples data shows nice improvements, especially when using a larger number of coefficients.

For financial data, on the other hand, the additional complexity of the new method does not pay off in significantly improved energy preservation,

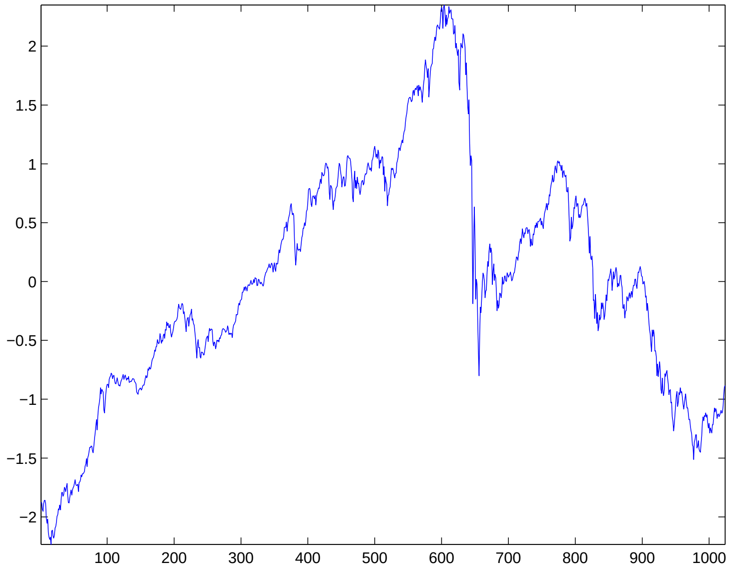

Figure 4: Example time series of sp500

k	**1**	**3**	**7**	**15**	**31**	**63**	**127**	**255**	**511**
buoy_sensor	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.2	0.7
cstr	0.0	0.0	0.0	1.4	2.2	1.9	0.6	0.1	0.0
eeg	2.9	6.0	5.0	6.5	6.5	7.3	3.5	5.3	3.1
ERP_data	1.8	1.4	13.5	11.5	12.5	3.0	1.4	0.3	0.0
evaporator	0.0	0.0	1.2	2.7	1.5	0.6	0.4	0.9	2.6
foetal_ecg	2.7	9.5	17.9	30.9	47.1	48.1	29.3	11.8	2.9
koski_ecg	1.7	3.4	2.6	0.2	0.5	0.5	0.0	0.0	0.0
network	1.3	3.1	5.0	7.8	13.8	19.8	25.7	26.6	18.5
pgt50_alpha	1.5	3.1	5.7	9.6	15.9	23.9	29.1	32.1	28.4
pgt50_cdc15	0.9	1.9	4.0	7.5	11.9	17.3	23.1	28.0	26.4
power_data	4.2	1.2	3.0	8.5	0.0	0.4	0.9	0.3	0.1
random_walk	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
spot_exrates	0.0	0.0	1.7	0.7	0.3	0.3	0.2	0.1	0.1
sp500	0.0	0.0	0.0	0.0	0.3	0.0	0.0	0.0	0.0
steamgen	0.0	0.2	0.6	0.0	0.1	0.3	0.2	0.2	0.1
synthetic_ctrl	5.0	13.9	17.4	7.3	10.6	7.6	8.4	8.2	7.2
tickwise	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 3: Gain in energy preservation with Haar DWT

see the examples *spot_exrates*, *sp500*, *tickwise*, and *random_walk* [2] . Other moderate performances are shown by the *buoy_sensor* and *steamgen* data.

For a list of the additional energy preserved by the optimal method using the Haar DWT, see Table 10 in the Appendix.

With the DFT feature extraction similar results were obtained. The best example is the *power_data* time series set with an improvement ranging from 10% to 47% for $k \in \{1, 3, 7, 15\}$ (see Table 4 and Figure 5). This data contains the power consumption of a research center averaged every 15 minutes for one year [30]. From an example of the set of times series in Figure 6 we can see clearly a very important high frequency component corresponding to the burst of consumption during the day compared to almost no consumption during the night. The traditional method of choosing the first few DFT coefficients would not select this frequency if k was not at least 15. This results in prototypes, that are not very similar to the original data they are representing and thus to a lot of false hits when querying by example.

Again the methods $adap_k$ and $each_k$ show almost the same performance. The additional energy preserved by the optimal method over the $best_k$ method is rather small compared to the improvement made. For $k = 1$ the remaining gap is about the same as the improvement achieved, for $k \in \{3, 7, 15\}$ far more energy preservation is gained than is left to the optimum.

k	$adap_k$	$each_k$	$best_k$	$first_k$	gain	left
1	24.5	24.3	14.9	4.9	**10.1**	9.6
3	45.3	44.7	34.7	11.6	**23.1**	10.6
7	67.6	67.2	52.2	25.4	**26.8**	15.4
15	85.7	85.4	78.3	31.2	**47.1**	7.4
31	95.0	94.9	91.6	87.5	**4.1**	3.4
63	98.1	98.1	97.5	95.9	**1.6**	0.7
127	99.2	99.1	98.7	98.6	**0.1**	0.5
255	99.7	99.6	99.4	99.3	**0.0**	0.3
511	99.9	99.9	99.8	99.7	**0.0**	0.2
1023	100.0	100.0	100.0	100.0	**0.0**	0.0

Table 4: Energy preservation with DFT for power_data

[2] random walk is commonly assumed to be very similar to stock price time series

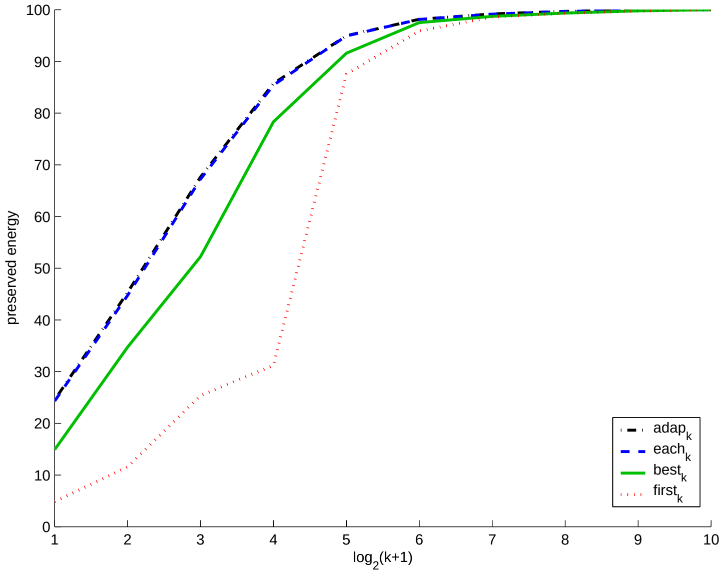

Figure 5: Energy preservation with DFT for power_data

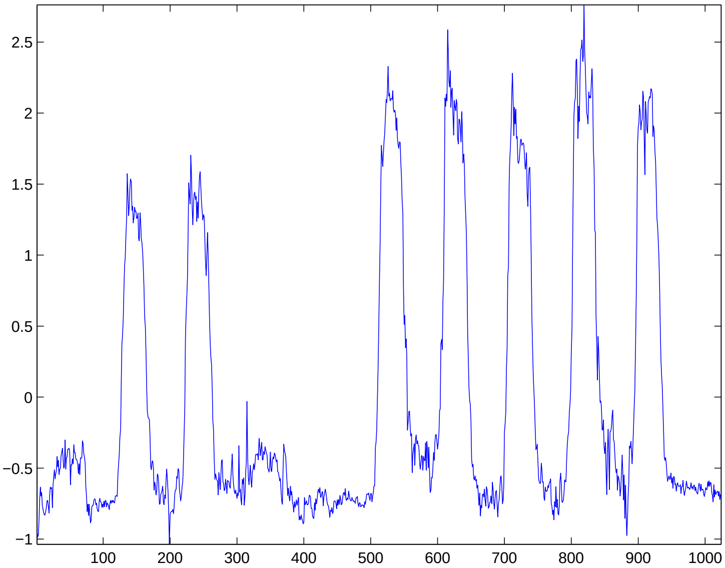

Figure 6: Example time series of power_data

Really bad examples for the proposed method using DFT are hard to find. Even the fincancial data sets show some improvement, at least for small k. The results for all time series data sets and the DFT are listed in Table 5. Again medical data performs well, while financial data performs comparatively poor.

k	**1**	**3**	**7**	**15**	**31**	**63**	**127**	**255**	**511**
buoy_sensor	6.6	3.6	0.1	0.9	3.1	3.7	0.5	0.2	0.3
cstr	0.0	3.8	3.2	1.4	0.6	0.8	0.1	0.1	0.0
eeg	7.0	9.0	18.8	17.8	22.0	18.4	13.5	13.6	11.5
ERP_data	5.2	14.7	6.3	3.2	4.7	5.1	0.8	0.8	0.0
evaporator	1.1	0.6	1.8	2.9	2.5	1.0	0.7	2.5	4.6
foetal_ecg	1.7	4.2	11.8	20.1	29.5	26.0	13.0	0.6	0.0
koski_ecg	3.8	6.7	5.9	3.4	5.2	1.5	0.0	0.0	0.0
network	1.3	2.4	1.5	2.0	2.9	3.3	3.8	3.2	2.9
pgt50_alpha	0.0	0.0	0.4	1.6	3.4	6.6	10.6	14.9	18.0
pgt50_cdc15	0.0	0.3	1.0	2.5	4.7	7.4	11.2	15.9	18.3
power_data	10.1	23.1	26.8	47.1	4.1	1.6	0.1	0.0	0.0
random_walk	20.4	4.2	1.4	0.5	0.2	0.1	0.0	0.0	0.0
spot_exrates	8.5	2.3	3.6	1.8	0.4	0.2	0.2	0.1	0.1
sp500	5.5	9.9	1.7	1.1	0.3	0.1	0.1	0.0	0.0
steamgen	4.6	9.2	0.5	0.3	2.3	0.3	0.1	0.1	0.0
synthetic_ctrl	33.4	39.7	38.3	3.5	3.8	3.2	5.0	8.0	10.6
tickwise	15.7	4.8	2.0	0.6	0.2	0.1	0.0	0.0	0.0

Table 5: Gain in energy preservation with DFT

For a list of the additional energy preserved by the optimal method using the DFT, see Table 11 in the Appendix.

More experiments were run with other Daubechies Wavelet bases (DB2, DB4, and DB12). The gains of the new method over the traditional method were usually even higher than with the Haar DWT or the DFT shown above. But the energy preservation was generally much worse for the signals used (in contrast to [22]), so the results are not listed here.

Several data sets from other applications were also not listed here, because only small data sets were available. It should be mentioned though, that the $best_k$ selection method was also very effective for the data sets *speech*, *earthquake*, and *robot_arm* from the TSDMA.

6.2 Clustering

One commonly used method to reveal unknown groups in data is clustering. When given a dataset with known classes, clustering can be used to test the feature extraction. The task is to find the true classes as good as possible, thus we used the classification error of the resulting partition as the the quality measure.

Since it is really the feature extraction we are testing, we used k-Means clustering with k equal to the number of true classes and initialized the cluster centers with one example of each class, to minimize the influence of the clustering algorihtm. Note that k-Means is biased toward spherical clusters, but since it was used to cluster the original data, as well as the transformed data this shouldn't cause any problems in the comparsion.

The classification errors in percent for the first_k and best_k methods are shown in Figure 7 for the DWT and Figure 8 for the DFT. Results for larger k were omitted, because the error for the level wise selection strategy started to rise again when using more coefficients. This means that using more coefficients only introduces noise and worsens the results. Note that clustering of the original data had a classification error of 11%.

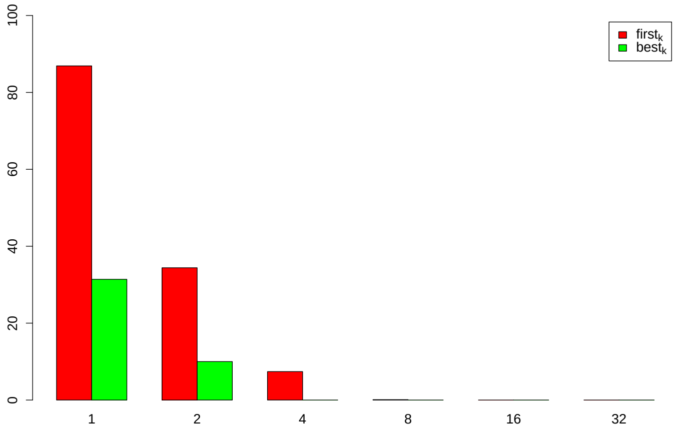

Figure 7: k-Means classification errors with Haar DWT

The proposed feature extraction works well. Especially for small k it outperformed the first_k selection by large margins. A perfect classification can be reached with the DWT and best_k with as few as 4 coefficients while

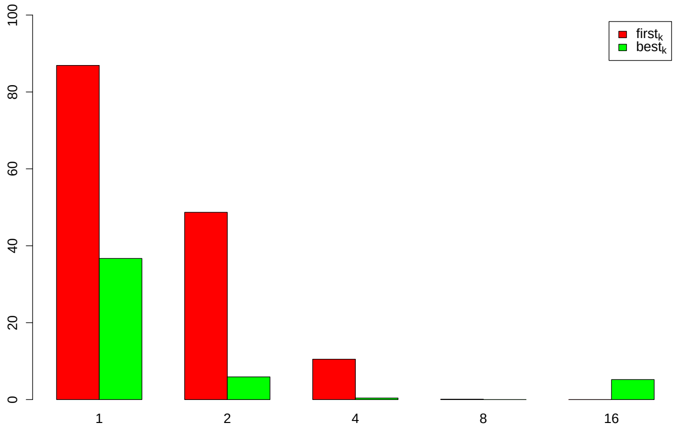

Figure 8: k-Means classification errors with DFT

the *first*$_k$ selection strategy still has an error of 7.4% and needs 16 coefficients to reach 0%.

6.3 Rule Generation

To convert groups, e.g. found by clustering, to knowledge interpretable by humans, rule generation algorithms can be used. We used the C5.0 [23] and the Sig* [28] algorithms on the same benchmark problem offering them an increasing amount of features to choose from, selected the *first*$_k$ and *best*$_k$ methods. Both algorithms were run with the default settings. Better classification results could probably be obtained by tuning the parameters for the problem at hand. Note, that we do not want to compare the two algorithms, neither in their classification performance nor their rule complexity. They are rather different in nature: Sig* is designed to produce more understandable rules while C5.0 uses a greedy search to minimize the number of decisions.

The numbers of features used were $k \in \{2, 4, 8, 16\}$. For $k = 1$ the classification error was rather high and for $k > 16$ the results stabilized, more features did not improve the results significantly. To check whether the generated rules can be compared, the classification errors of the training set and the test set are listed. But the main focus is on the complexity of the rule set, measured simply by the number of rules and the total number of conditions within these rules.

The results for the C5.0 algorithms applied to the Haar DWT are listed in Table 6. Given $k = 2$ features choosen with the $best_k$ method actually produced a more complex rule set than using the $first_k$ method (3 rules and 11 conditions more). But at the same time it achieves half the error rate so the rule sets can't be compared in a fair manner. With 4 features or more all errors are zero for both methods. For $k = 4$ the $best_k$ method has a smaller rule set (6 rules and 25 conditions less) than the $first_k$ method. Thus here the proposed method leads to rules better interpretable by humans. For $k > 4$ both method always create the same number of rules and conditions.

		Classification error		Rule set complexity	
Method	k	Training	Test	Rules	Conditions
$first_k$	2	25.5%	27.8%	9	16
$best_k$	2	10.5%	12.5%	12	27
$first_k$	4	0%	0%	**17**	**57**
$best_k$	4	0%	0%	**11**	**32**
$first_k$	8	0%	0%	11	33
$best_k$	8	0%	0%	11	33
$first_k$	16	0%	0%	11	33
$best_k$	16	0%	0%	11	33

Table 6: C5.0 rules with Haar DWT

When using C5.0 on the DFT features (see Table 7) the difference between the two competing methods is even larger. For $k = 2$ the rule set is much smaller *and* has zero error compared to 50% error of the $first_k$ method on the test set. For $k = 4$ the errors are the same, but again $best_k$ produces less rules and less conditions. As above the results stabilized for larger k.

The results for the Sig* algorithms listed in Table 8 and Table 9 also show how the $best_k$ method can produce simpler rule sets with a higher classification performance. Using the DWT and $k = 2$ the rule sets are comparable in size, but $best_k$ has a much lower error rate on both the training and the test set. Using 4 features the $best_k$ method achieves less than half of the error rates of the $first_k$ method using 4 rules and 19 conditions less. For $k = 8$ the error rates are comparable, but the $first_k$ method has a more complex ruleset. Sig* applied to the DFT features shows more excellent performance of $best_k$ for small values of k. Similar to C5.0 the results stabilize for larger k stabilize.

		Classification error		Rule set complexity	
Method	k	Training	Test	Rules	Conditions
$first_k$	2	28.2%	50.8%	**28**	**79**
$best_k$	2	0%	0%	**12**	**32**
$first_k$	4	0%	0%	**14**	**47**
$best_k$	4	0%	0%	**10**	**29**
$first_k$	8	0%	0%	10	29
$best_k$	8	0%	0%	10	29
$first_k$	16	0%	0%	10	28
$best_k$	16	0%	0%	10	28

Table 7: C5.0 rules with DFT

		Classification error		Rule set complexity	
Method	k	Training	Test	Rules	Conditions
$first_k$	2	38.0%	41.0%	20	52
$best_k$	2	28.0%	27.0%	17	50
$first_k$	4	33.0%	30.0%	**18**	**64**
$best_k$	4	12.0%	13.0%	**14**	**45**
$first_k$	8	13.0%	17.0%	**15**	**60**
$best_k$	8	14.0%	18.0%	**12**	**53**
$first_k$	16	21.0%	24.0%	13	79
$best_k$	16	21.0%	24.0%	13	80

Table 8: Sig* rules with Haar DWT

		Classification error		Rule set complexity	
Method	k	Training	Test	Rules	Conditions
first_k	2	52.0%	55.0%	**34**	**94**
best_k	2	11.0%	14.0%	**18**	**48**
first_k	4	31.0%	28.0%	**21**	**72**
best_k	4	10.0%	12.0%	**13**	**44**
first_k	8	13.0%	15.0%	12	54
best_k	8	13.0%	14.0%	13	54
first_k	16	18.0%	22.0%	12	77
best_k	16	19.0%	23.0%	11	73

Table 9: Sig* rules with DFT

7 Discussion

The new method for the selection of features from a set of time series based on DWT and DFT is superior in energy preservation to the traditional method. Large improvements were reported for many datasets, an additional 48% of the original energy was preserved for some medical signals. On other data sets, e.g. fincancial data, only slight improvements could be observed. This is because for signals with dominant low frequencies the traditional method already is very close to the optimum, so there is hardly anything to improve.

While even more energy could be preserved by using the largest coefficients of each time series individually, this would introduce storage overhead, bookkeeping problems for the distance computation and reduce the suitability of the feature set to produce interpretable rules.

Any data mining algorithm should profit form the improved energy preservation, because the features represent time series more similar to the original time series. Clustering, classification and rule generation algorithms will profit from the more complex shapes representable by keeping coefficients at arbirary positions.

For time series indexing the new method will produce less false hits for a query and reduce the time neccessary for post processing. Even though not all time series might be available to begin with, an index could be built with a representative subset first and later be rebuilt whenever a lot of new data was added. It would be interesting to implement the new method in combination with time series indexing methods to see how the improved energy preservations translates into better values of precision and recall and reduces the number of disk accesses needed.

While a strategy for choosing the best coefficients for a time series set was given, the question, how many coefficients should be chosen, remains open. One could use the elbow criterium, that is common pratice when using principal component analysis. For indexing this is sometimes answered by the number of dimensions an indexing method can handle efficiently. Popivanov *et. al.* [22] investigate the interaction of the wavelet base, the number of extracted features and the indexing method used. For other data mining tasks no such restrictions apply.

For classification, generic feature selection algorithms based on cross-validation could also be used. Our method produces more similar prototypes rather than optimizing classification performance for a particular problem. Further, our method works with unsupervised methods as well as supervised methods.

For clustering, the proposed selection strategy could be combined with the recently introduced incremental k-Means [20] algorithm. Here, the more fine grained selection possibilities of our approach might be beneficial in addition to the better energy preservation.

Note that using k-Means clustering is in general problematic because the number of clusters needs to be known in advance, unless using some heuristic for adding and merging centers. Using Emergent Self Organizing Maps (ESOM) with the U-Matrix and the P-Matrix [29] is a better way to investigate the cluster structure of a high dimensional data set without requiring prior knowledge. It can also deal with points not belonging to any cluster naturally. But the manual classification step neccessary when using ESOM is problematic when testing classification performance automatically, as needed here.

While in our experiments either DWT or DFT were used, hybrid techniques are also possible. The short time FT could be used to extraxt regular, texture like features, while the DWT extracts isolated, singular features from the remaining signal.

8 Conclusion

A new method for selecting features from a set of time series based on DWT and DFT was presented. Selecting the coefficients by their mean energy was shown to be optimal with respect to energy preservation under the constraint of choosing the same subset of coefficients for all time series. A large improvement in energy preservation over the traditional method of choosing the first coefficients was observed for many real life data sets. The performance was also fairly close to the optimal procedure of choosing an adaptive number of coefficients from different position for each time series. This method should only be chosen if performance and interpretability are of minor concerns.

The better energy preservation of the proposed method will improve the application of data mining algorithms like clustering, rule generation, classification and indexing. The new method was shown to improve clustering and reduce the size of classification rule sets on a benchmark data set.

Acknowledgements

Thanks go to Eamon Keogh and Jessica Lin for providing the datasets, Stefan Dahlke for useful discussion on wavelets and Alfred Ultsch for discussion on data mining.

References

[1] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. *Lecture Notes in Computer Science*, 1973:420–??, 2001.

[2] R. Agrawal, K. I. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in times-series databases. In *Proceedings of 21rst International Conference on Very Large Data Bases*, pages 490–500, 1995.

[3] Rakesh Agrawal, Christos Faloutsos, and Arun N. Swami. Efficient Similarity Search In Sequence Databases. In D. Lomet, editor, *Proceedings of the 4th International Conference of Foundations of Data Organization and Algorithms (FODO)*, pages 69–84, Chicago, Illinois, 1993. Springer Verlag.

[4] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and rectangles.

In *Proceedings of ACM SIGMOD Int'l. Conf. on Management of Data*, pages 322–331, 1990.

[5] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is "nearest neighbor" meaningful? *Lecture Notes in Computer Science*, 1540:217–235, 1999.

[6] Kin-Pong Chan and Ada Wai-Chee Fu. Efficient time series matching by wavelets. In *ICDE*, pages 126–133, 1999.

[7] Kin-Pong Chan, Ada Wai-Chee Fu, and Clement T. Yu. Haar wavelets for efficient similarity search of time-series: With and without time warping. *IEEE Transactions on Knowledge and Data Engineering*, pages 686–705, 2003.

[8] I. Daubechies. *Ten lectures on Wavelets.* SIAM, 1992.

[9] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. Fast subsequence matching in time-series databases. In *Proceedings 1994 ACM SIGMOD Conference, Mineapolis, MN*, pages 419–429, 1994.

[10] T. C. Fu, F. L. Chung, V. Ng, and R. Luk. Pattern discovery from stock time series using self-organizing maps, 2001.

[11] Pierre Geurts. Pattern extraction for time series classification. *Lecture Notes in Computer Science*, 2168:115–127, 2001.

[12] Dina Q. Goldin and Paris C. Kanellakis. On similarity queries for time-series data: Constraint specification and implementation. In *Proceedings of the 1st International Conference on Principles and Practice of Constraint Programming (CP'95)*, Cassis, France, 1995. Springer Verlag.

[13] C. Goutte. On clustering fMRI time series, 1998.

[14] Mohammed Waleed Kadous. Learning comprehensible descriptions of multivariate time series. In *Proc. 16th International Conf. on Machine Learning*, pages 454–463. Morgan Kaufmann, San Francisco, CA, 1999.

[15] K. Kalpakis, D. Gada, and V. Puttagunta. Distance measures for effective clustering of ARIMA time-series. In *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM'01), San Jose, CA*, pages 273–280, 2001.

[16] E. Keogh. The UCR Time Series Data Mining Archive, *http://www.cs.ucr.edu/˜eamonn/tsdma/index.html*, 2002.

[17] Eamonn Keogh and Shruti Kasetty. On the need for time series data mining benchmarks: A survey and empirical demonstration.

[18] J. Lin, E. Keogh, and W. Truppel. Clustering of streaming time series is meaningless. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, San Diego, CA. June 13*, 2003.

[19] S. G. Mallat. *A Wavelet Tour of Signal Processing.* Academic Press, 1999.

[20] Eamonn Keogh Michail Vlachos, Jessica Lin and Dimitrios Gunopulos. A wavelet-based anytime algorithm for k-means clustering of time series, 2003.

[21] Yang-Sae Moon, Kyu-Young Whang, and Woong-Kee Loh. Efficient time-series subsequence matching using duality in constructing windows. *Information Systems*, 26(4):279–293, 2001.

[22] Ivan Popivanov and Renee J. Miller. Similarity search over time-series data using wavelets. In *ICDE*, page 0212, 2002.

[23] J.R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufman Publishers, 1993.

[24] Davood Rafiei and Alberto O. Mendelzon. Similarity-based queries for time series data. pages 13–25, 1997.

[25] Davood Rafiei and Alberto O. Mendelzon. Efficient retrieval of similar time sequences using DFT. In *FODO*, 1998.

[26] Juan J. Rodríguez, Carlos J. Alonso, and Henrik Boström. Learning first order logic time series classifiers, 2000.

[27] O. Tim, L. Firoiu, and P. Cohen. Clustering time series with hidden markov models and dynamic time warping, 1999.

[28] A. Ultsch. The integration of neural networks with symbolic knowledge processing. In *New Approaches in Classification and Data Analysis, Springer Verlag*, pages 445–454, 1994.

[29] A. Ultsch. Maps for the visualization of high dimensional data spaces. In *WSOM*, 2003.

[30] Jarke J. van Wijk and Edward R. van Selow. Cluster and calendar based visualization of time series data. In *INFOVIS*, pages 4–9, 1999.

[31] Yi-Leh Wu, Divyakant Agrawal, and Amr El Abbadi. A comparison of DFT and DWT based similarity search in time-series databases. In *CIKM*, pages 488–495, 2000.

Appendix

k	**1**	**3**	**7**	**15**	**31**	**63**	**127**	**255**	**511**
buoy_sensor	11.1	11.8	9.3	9.3	10.5	11.0	9.2	7.4	4.4
cstr	17.2	24.1	20.6	10.3	8.0	3.9	1.7	0.5	0.1
eeg	6.7	10.0	11.8	12.4	11.0	9.2	10.1	9.8	6.5
ERP_data	14.8	27.9	33.9	25.5	8.3	3.6	1.7	1.1	0.5
evaporator1	5.4	10.6	15.2	17.9	17.8	13.1	8.0	5.4	3.5
foetal_ecg	7.0	12.7	21.2	27.8	22.8	13.3	6.1	1.9	0.5
koski_ecg	11.5	25.2	40.3	46.1	42.4	25.9	10.2	2.9	0.6
network	13.0	24.1	34.1	41.6	45.2	43.1	34.0	20.3	7.4
pgt50_alpha	1.6	3.8	6.2	8.8	11.2	13.5	17.1	19.3	14.9
pgt50_cdc15	1.6	3.4	5.7	8.5	12.0	16.2	20.9	23.3	17.3
power_data	6.0	14.2	26.0	33.7	4.1	4.7	2.2	0.8	0.3
random_walk	8.6	9.9	6.4	3.5	2.0	1.0	0.6	0.3	0.2
spot_exrates	20.3	16.6	7.6	3.3	1.5	0.9	0.4	0.3	0.1
sp500	15.2	7.0	5.4	3.4	1.9	0.9	0.5	0.2	0.1
steamgen	15.6	21.8	21.2	14.4	8.3	5.0	2.4	1.1	0.4
synthetic_ctrl	3.5	4.5	7.2	7.6	4.2	6.3	9.2	11.4	9.2
tickwise	12.1	11.4	7.7	4.7	2.8	1.5	0.8	0.4	0.2

Table 10: Additional energy by optimal method with Haar DWT

k	**1**	**3**	**7**	**15**	**31**	**63**	**127**	**255**	**511**
buoy_sensor	10.5	10.9	9.5	9.1	8.7	6.6	5.5	5.2	3.7
cstr	17.7	24.6	15.1	4.5	4.5	2.5	0.6	0.3	0.2
eeg	10.9	13.8	13.7	12.7	9.1	9.2	8.7	5.6	-0.0
ERP_data	5.1	10.5	15.2	12.2	5.6	2.1	0.6	0.3	0.0
evaporator1	7.1	11.6	17.8	21.6	21.1	13.2	4.6	3.7	2.7
foetal_ecg	6.2	9.1	11.0	14.7	18.0	17.3	8.8	1.4	0.3
koski_ecg	9.4	16.9	20.3	20.1	21.2	13.9	1.2	0.1	0.0
network	1.1	2.5	5.1	8.4	12.9	18.0	21.4	20.4	14.0
pgt50_alpha	1.0	2.2	4.1	7.0	11.4	16.9	23.4	28.3	22.7
pgt50_cdc15	0.7	2.0	3.9	6.7	11.0	17.1	24.4	29.5	23.6
power_data	9.6	10.6	15.4	7.4	3.4	0.7	0.5	0.3	0.2
random_walk	11.9	10.5	5.8	3.2	1.6	0.9	0.4	0.2	0.2
spot_exrates	26.2	19.6	8.7	2.9	1.3	0.5	0.3	0.2	0.1
sp500	23.2	7.1	4.5	2.9	1.4	0.6	0.3	0.2	0.1
steamgen	18.2	24.6	20.3	7.5	3.5	1.8	1.2	0.7	0.2
synthetic_ctrl	1.9	3.3	2.3	2.7	5.0	8.9	12.1	12.6	7.4
tickwise	14.8	11.4	6.1	3.1	1.7	0.8	0.4	0.3	0.2

Table 11: Additional energy by optimal method with DFT