Raport Badawczy
Research Report

RB/51/2012

An approach for reduction of dimension of data series

M. Krawczak, G. Szkatuła

Instytut Badań Systemowych
Polska Akademia Nauk

Systems Research Institute
Polish Academy of Sciences

POLSKA AKADEMIA NAUK

Instytut Badań Systemowych

ul. Newelska 6

01-447 Warszawa

tel.: (+48) (22) 3810100

fax: (+48) (22) 3810105

Kierownik Zakładu zgłaszający pracę:
Prof. zw. dr hab. inż. Janusz Kacprzyk

Warszawa 2012

An approach for reduction of dimension of data series

Maciej Krawczak and Grażyna Szkatuła

Systems Research Institute, Polish Academy of Sciences, Newelska 6, Warsaw, Poland
e-mail: {krawczak, szkatulg}@ibspan.waw.pl

Abstract

Many methods of reducing dimensionality of data series have been introduced over the past decades. Some of the methods introduce a symbolic representation form of the original data series, but obtained dimensionality reduction is not significant. In this paper, we introduce a new approach SEAA to reduce dimensionality of multidimensional data series. The approach creates a nominal (symbolic) representation of the original data series and considerably reduces their dimensionality. The approach consists of several steps, and each step gives a new data series representation as well as dimension reduction. The approach is based on the concept of data series envelopes and principal components-like - called here 'essential attributes' - generated by a multilayer neural network. The essential attributes are represented by outputs of hidden layer neurons. Next the real values essential attributes are nominalized, and in this way nominal data series representation is obtained. It must emphasized that by a data series we mean a time series or pseudo-time series while SEAA generates a set of nominal values of attributes which describe the compressed representation of original data series, and a fixed permutation of nominal attributes should be considered. The nominal attributes are synthetic, and there is not any physical interpretation of them, but they still retain important features of the original data series. Experimental validation of the proposed dimension reduction was carried out for classification and clustering tasks. The calculations have shown that even deployment of large reduction of dimensionality causes the new representations to preserve information about the data series characteristics and retain information sufficient to their proper classification and clustering.

Keywords: Data series; Nominal attributes; Dimension reduction; Envelopes; Essential attributes

1. Introduction

The term "data series" is often used to refer to any data set with one, independent time variable. Data series arise in many areas, such as medicine, finance, industry, climate etc., and the generated data must be registered, stored, transmitted, and then analysed. The majority of data series research focuses on the following problems:

- *indexing* (e.g. Keogh, Chakrabarti and Pazzani, 2001),
- *clustering* (e.g. Keogh and Pazzani, 2001; Wu and Chang, 2004; Krawczak and Szkatuła, 2010c, 2011),
- *classification* (e.g. Nanopoulos, Alcock, Manolopoulos, 2001; Krawczak and Szkatuła, 2010a, b, 2011), Wang (2010),
- *summarization* (e.g. Lin, Keogh, Patel and Lonardi, 2002), and
- *anomaly detection* (e.g. Shahabi, Tian and Zhao, 2000).

Due to a huge amount of data, different kinds of data series representations were developed. In the literature one can find specialized algorithms dealing with such problems, including decision trees (Rodriguez and Alonso, 2004), neural networks (Nanopoulos, Alcock and Manolopoulos, 2001), bayesian classifiers (Wu and Chang, 2004), etc. Some representations are general enough to be used in the mentioned problems, and some are rather specialized, meant for prescribed applications. It is worth to mention that there is an increasing interest in data series mining, e.g. Xi, Keogh, Shelton, Wei, Ratanamahatana, 2006. It is said that time series or data series mining is considered as one of the tenth challenging problems in data mining (Yang and Wu, 2006; Fu, 2011).

There are some problems in treatment of high dimension data, including the curse of dimensionality and the meaningfulness of the similarity measure in high dimension space. For any point in a high dimensional space, the expected gap between the Euclidean distance to the closest neighbor and to the farthest point decreases while the dimensionality grows. This phenomenon may cause many data mining tasks to render ineffective and fragile (Beyer et al., 1999). Also, the data mining methods require high computational cost applying very large data sets. This obstacle is sometimes known as the "curse of dimensionality" (Elder and Pregibon, 1996).

In most of the above data series mining problems, there is a necessity of reducing dimensionalities and forming new data series representations. It is required that the new representation preserve sufficient information for solving above data series problem with satisfactory accuracy. Reducing dimensionality (either the number of data point or the number of records), can effectively cut this computational cost.

Therefore, the reduction of the original series' dimensionality is crucial because dimensionality reduction decreases the cost, increases the performance, or reduces redundant dimensions. Reduction of dimensions can be divided into major problems: *attribute selection*, *attribute extraction* and *record selection*. The process of extracting the most important attributes or formation of the new attributes based on the original set reduces the dimensionality of the data. Also, some records or examples may better support the learning process of data mining than others (Maimon and Rokach, 2010). Note that lossy compression methods can always achieve higher compression rates but involve a trade-off between compression rate and error, and the goal is to achieve the best ratio between compression rate and error.

There are many approaches to dimensionality reduction and similarity searches of data series in large databases (Tak-chung Fu, 2011). A data series of arbitrary length M can be reduced to another representation of data series of length K, $K<M$. The simplest method is *sampling* (Astrom, 1969) in which a rate of M/K is used. The method, however, does not retain the shape of compressed time series if the sampling rate is too low.

Piecewise approximation methods divide the data series into segments and approximate each segment using some functions. An enhanced method is to use the average value of each segment to represent the data point in the new, compressed representation. One of these methods is based on *piecewise constant approximation* (PCA), also known as *piecewise aggregate approximation* (PAA). In their papers, (Yi and Faloutsos, 2000) and (Keogh et al., 2000) proposed to divide each data series into segments of equal length and to use the average value of each segment to represent the latter PAA. Keogh at. al. (2000, 2001) have also proposed an extended version called an *adaptive piecewise constant* approximation (APCA), where the segments length is not fixed. Instead of using the average value to represent each segment, other methods are proposed, e.g. Lee at al. (2003) proposed to use the segmented sum of variation (SSV) to present each segment, while Ratanamahatana at al. (2005) and Bagnall at al. (2006) proposed a bit level approximation.

There are other methods to approximate a time series by straight lines; for example, by linear interpolation (Keogh, 1997), (Keogh and Smyth, 1997), (Smyth and Keogh, 1997), or linear regression (Shatkay and Zdonik, 1996).

Furthermore, preserving the salient points seems to be a promising method, such perceptually important points (PIP) were first introduced by Chung et. al. (2001).

The idea of upper and lower envelopes of data series was introduced by Krawczak and Szkatuła (2010a, 2010b) and is worth to be considered and used.

Representing data series in the transformation domain is another approach. One of the popular transformation techniques is the *discrete Fourier transforms* (DFT) (Faloutsos, Ranganathan and

Manolopulos, 1994) and the *discrete wavelet transform* (DWT) (Chan and Fu, 1999). *Principal component analysis* (PCA) is a popular multivariate technique using statistical methods (Yang and Shahabi, 2005), (Yoon et al., 2005). Another methods use hidden *Markov models* (HMMs), (Azzouzi and Nabney, 1998). Many of the approaches use different *indexing method.*

One important feature of all the above approaches is that they operate on real values. Another common family of approaches converts the numerical time series to symbolic form. The simplest method is discretizing the time series into segments and converting into a symbol (Yang and Zhao, 1998; Yang et. al., 1999). Also a symbolic PAA technique called *symbolic aggregate approximation* (SAX) was introduced by Lin, Keogh and Lonardi (2007). They convert the result from PAA to symbol string. Two parameters must be specified for the conversion: the length of subsequence and alphabet of symbols used. SAX preserves the general shape of the original time series.

In this paper, we propose a new approach: SEAA - *symbolic essential attributes approximation* for gradual reduction of dimension of multidimensional data series. Our approach allows a data series of arbitrary length M to be reduced to arbitrary length K, where $K << M$. For symbolic representation of data series, we use alphabet of finite size, $R >= 3$. Our approach differs from other methods known in the literature. In general, these methods give compressed representation of data series which preserve the time order of the original data series, while in our case we obtain a set of nominal values which preserves characteristics of the original data series and only a fixed permutation of the attributes can be considered.

The attributes are synthetic and there is no physical interpretation of them, but they still hold the most important features of the original data series. Although the approach does not preserve the general shape of the original time series, it contains enough information for their proper classification and clustering. The proposed methodology consists of several steps in which considerably dimensionality reduction is performed. Compression ratio at each step is determined in an experimental way and depends on the considered data.

The remaining part of this paper is organized as follows: Section 2 presents the clarification of the proposed approach, in Section 3 we present description of the methodology and the data series dimension reduction is described in details. Practical presentation of the proposed approach was carried out for the database available at the Irvine University of California in Section 4. Using the attributes with nominal values as aggregated data series representation verification of the proposed approach was carried out for two data series mining problems, namely classification and clustering. We consider classification and clustering problem, because they are among of the most common data mining problems. We have made calculations on compressed data in order to determine whether they contain enough information to their proper classification and clustering. In Section 5 there are examples which show the efficiency of the proposed methodology. In Appendix A, the basic elements of the used extraction of decision rules are presented. We used the method of creating the minimal set of rules successively for each class developed by Szkatuła (1995, 2002), Kacprzyk and Szkatuła (1999, 2002, 2005a, 2005b, 2010). Appendix B gives basic elements of the adopted algorithm applied for clustering (Krawczak and Szkatuła, in preparation).

2. Clarification of the approach

Our work is motivated by the observation that using combination of several methods for data series dimensionality reduction is more universal than using a single method. It seems that instead of using one method with a large loss of information, it is much more efficient to compress several methods in which information is reduced gradually with partial little loss of information.

Our approach allows a data series of arbitrary length M to be reduced to another representation of data series of length K, where $K << M$. We propose a new approach which changes the real values data series representation into a new nominal representation of data series. During this representation changes there is significant reduction of data series dimensionalities. The propose methodology consists of several steps, during which dimensionality reduction is obtained and compression ratio at each step is determined. It must emphasized that the data series is a time series

or pseudo-time series while SEAA generates a set of nominal value attributes arranged according to one fixed permutation of attributes for all considered objects.

The developed methodology is shown in Fig.1.

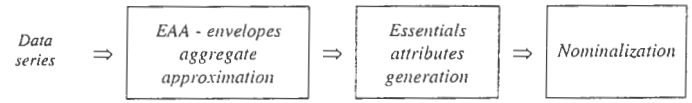

Fig. 1. The approach scheme of SEAA - *symbolic essential attributes approximation*

Before starting the dimension reduction of the data series, pretreatment of data should be performed and each series should be normalized to have mean equal zero and standard deviation equal one. Let us denote the original data series of arbitrary length M and indexed by n by the following vector form $[x_1(n), x_2(n), ..., x_M(n)]$, $n = 1, ..., N$. Concept of each step of SEAA methodology is briefly described below.

2.1. Envelopes aggregate approximation

First, we introduce a piecewise approach for representing a data series, introduced by Krawczak and Szkatuła (2010a, 2010b). Following the idea borrowed from the signal processing theory we developed piecewise constant upper and lower approximation of data series. We applied the piecewise constant functions, also called step functions, with equal length of steps. The length of steps is denoted as m-step, meaning that m succeeding samples of data series constitute one step. Upper and/or lower approximation are developed, and then aggregated. The parameter m describes the rank of time series dimension reduction. It means for a given data series of the length M we obtain the reduced length $\left\lfloor \frac{M}{m} \right\rfloor$, where $m << M$.

Selection of the proper values of the m parameter is of crucial importance because this value has strong influence on quality of time series representations. The m parameter should be adjusted in an experimental way. It is required that the dimension reduction of data needs to be rational in the following sense: from one point of view the value of m should be as large as possible then the dimension reduction is significant, but at the same time increasing value of m causes losing of information involved in data series. The problem of adjusting the value of the parameter m seems to be no trivial at all, and much experimental investigation should be performed.

The aggregated envelopes give a first new data series representation thereby the first reduction of dimensionality as the following vector $[y_1(n), y_2(n), ..., y_{\lfloor \frac{M}{m} \rfloor}(n)]$, $n = 1, ..., N$.

2.2. Essential attributes generation

In the next step, we extract features of the considered data series in order to obtain the further dimension reduction. It is assumed that a feature is an identified variable which efficiently captures the information involved in the aggregated envelopes, and by implication involved in the original data series. The idea of using features is motivated by common belief that description of each multidimensional data series may be redundant. There are several features such as *Lyapunov exponents*, *ARMA* models, *wavelet transform*, *correlation dimension*, *statistical moments* or *principal components analysis* for data series analysis, e.g. (Oja, 1992), (Guyon, Gunn, Nikravesh, Zadeh, 2005). Here, in our approach we will exploit Cybenko's theorem (Cybenko, 1989) as well as nonlinear principle component analysis and auto-associative neural networks. Cybenko's theorem states about a function approximation, while nonlinear principle component analysis determines mapping from $\left\lfloor \frac{M}{m} \right\rfloor$-dimensional space of aggregated envelopes to E-dimensional space of components. A three layer auto-associative neural network can perform an identity mapping, where the network outputs are enforced to equal the network inputs with some accuracy, and the

features are represented by outputs of the second hidden layer neurons. Here we will name the features as *essential attributes*, and the assumed neural network architecture requires that the number of second hidden layer neurons is remarkable smaller than the dimension of aggregated envelopes, i.e. $E << \left\lfloor \frac{M}{m} \right\rfloor$. Under such assumption, an auto-associative neural network works as a devise for data compression and decompression, but here we put our attention to the first functionality of the network - in compression - to obtain the essential attributes, while decompression part is necessary to adjust neural network weights to keep good quality of compression/decompression.

We used multilayer feedforward neural networks to reconstruct the input data by the network output. To perform this task efficiently, such neural networks learn interrelationships among the input variables. When the network is trained successfully, a small number of "hidden neurons" is sufficient to reconstruct the input values as the network outputs. This way the data are compressed to a form represented by data of lower dimensions. The outputs of the hidden layer neurons constitute the essential attributes and the number of them E is adjusted, where $E << \left\lfloor \frac{M}{m} \right\rfloor$. Selection of the proper values of the number of generated essential attributes E is of crucial importance because their values have strong influence on quality of data series representations. The problem of choosing the value of E (i.e. number of essential attributes) must be overcome in some experimental way shown in Section 4.2. It should be emphasized that original data as well as the aggregated envelopes data representation have a form of series of time, and the order of samples is natural and of crucial importance. Meanwhile the essential attributes generated by the designed neural network have a set form where the order of elements is meaningless. However the essential attributes for all data must be generated for one chosen permutation of elements of the set of the essential attributes. This way we obtained another representation of the original data series, and the length E of the new representation indicates additional dimension reduction of data series representation. Now the data series representation described by the essential attributes has a set form $\{b_1(n), b_2(n), ..., b_E(n)\}$, $n = 1, ..., N$.

The essential attributes can be used directly or can be modified, or on the base of the essential attributes it is possible to generate a new set of attributes. The main reason to generate the new attributes is to express hidden relationships between individual attributes. These new attributes can be obtained in various ways, generally it is said that the new attributes are some functions of the original ones (Matheus, Rendell, 1989), (Wnek, Michalski, 1994). Often there are used functions like maximum value, minimum value, average value, etc. or some arithmetic operators including: +, -, * and integer division, and so on.

In our approach, the new attributes are calculated as rearrangements of differences of the original essential attributes. This way we slightly enlarge dimensionality of the data series representation, but in the same time we provided, in some sense, the distances between the essential attributes, which may be particularly important in the task of clustering. The new set of attributes is denoted as $\{c_1(n), c_2(n), ..., c_K(n)\}$, $n = 1, ..., N$, where the number of new attributes K is adjusted, $K \geq E$.

2.3. Attributes nominalization

The data mining methods often involve numeric data (either discrete or continuous). However, there exist many methods that are designed for data which attributes can have only a small number of possible values (nominal or ordinal). Even for algorithms that can directly deal with numeric data, learning is often less efficient and less effective, therefore nominalization is recommended.

Also, in many applications, the Euclidean distance does not behave properly in measuring the similarities between data series, especially when shifts on the time axis appear. After the data series has been transformed into a symbolic representation, we can treat each time series as a string and apply similarity measures in the string domain; these measures usually do not have the limitations of Euclidean distance.

It is worth to mention that the essential attributes generated by an auto-associative neural network are represented as real values and each such a value requires e.g. 32 bits to be stored or compute. Meanwhile, symbolic representation of data gives additional dimension reduction

of data representation. For instance, let the range of values of each attribute is divided into 8 parts then only 3 bits (instead of 32) are sufficient for storing information about a symbolic value of an attribute. Such data representation measured in bits is very important in data transmission.

Nominalization as a process of conversion of numeric data into sequential symbolic data plays an important role in data mining and knowledge discovery. It relies on dividing the real value range of attributes into a number of intervals, and next to assign nominal codes to each interval. There are some well known groups of methods used for nominalization (Maimon, Rokach, 2010), for example: *division of equal width intervals* and *equal frequency intervals* method, the *discretization based on statistical tests*, *entropy based discretization* and *methods with applying the dynamic programming*. All these methods can be treated as heuristic for discretization of data, and experiments show that none of them is significantly better than others, and the choice of method depends heavily on data considered and type of considered problem. It is unrealistic to pursue a universally optimal nominalization approach.

Note that in our approach there is no physical interpretation of the essential attributes and transformed essential attributes, they take values from some set of real numbers, and implicit relationships between essential attributes are very important. So, the nominalization of a set of values for all attributes should be considered at the same time but not individually. The symbolic replacement is done in such a way that the common range of the all attributes is divided into some sub-ranges; it means that the division is the same for each attribute thus each nominal value has the same interpretation for each attribute. We propose to use a particular and simple method called equal width interval discretization. Choose another method of discretization requires further study and remains an open question. It seems that such nominalization should express implicit relationships between individual attributes.

Thus, real values of the attributes are replaced by nominal values constituting the new data series representation $\{a_1(n), a_2(n), ..., a_K(n)\}$, $n = 1, ..., N$.

In the next section each step of our approach of the data series dimension reduction SEAA is described in details. Will be discussed the concept of envelopes, extracting essential attributes, and nominalization of the attributes.

3. Description of methodology

3.1. Envelopes aggregate approximation

To reduce the data series of length M, each data series is divided into equal sized intervals of length m. The maximum and minimum value of the data falling within each interval is calculated and these values become the data new representation. The representation can be visualized as an attempt to approximate the original data series with a linear approximation of intervals. The exemplary upper and lower approximation concept of an exemplary data series for $m = 4$ and $M = 20$ is visualized in Fig. 2 a). Next m succeeding equal values are treated as a single value and such envelopes values from Fig. 2 a) are visualized in Fig. 2 b) as aggregated envelopes.

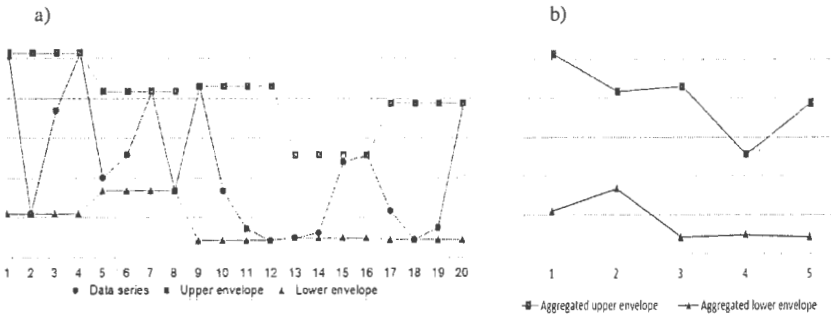

Fig. 2. a) The *4*-step upper and lower approximations, b) The aggregated *4*-step envelopes.

This way, we obtained the new representation of the data series, the new representation is described by either upper or lower envelopes, or by both kinds of the envelopes. Below we will discuss in details construction of envelopes.

Let us consider the normalized data series described in the following way:

$$\{x_k(n)\}_{k=1}^{k=M} = [x_1(n), x_2(n), \ldots, x_M(n)],\ n = 1, 2, \ldots, N. \tag{1}$$

The so-called m-step upper and m-step lower envelopes, $m << M$, constitute a kind of approximations of (1). The m-step upper approximation (upper envelope) of a data series (1) is denoted by $\{x_k^U(n)\}_{k=1}^{k=\left\lfloor \frac{M}{m} \right\rfloor m}$, and has the following form:

$$\begin{aligned}
&x_1^U(n) = \max\{x_1(n), x_2(n), \ldots, x_m(n)\} \\
&\ldots \\
&x_m^U(n) = \max\{x_1(n), x_2(n), \ldots, x_m(n)\} \\
&x_{m+1}^U(n) = \max\{x_{m+1}(n), x_{m+2}(n), \ldots, x_{2m}(n)\} \\
&\ldots \\
&x_{2m}^U(n) = \max\{x_{m+1}(n), x_{m+2}(n), \ldots, x_{2m}(n)\} \\
&\ldots \\
&x_{\left\lfloor \frac{M}{m} \right\rfloor m-m+1}^U(n) = \max\{x_{\left\lfloor \frac{M}{m} \right\rfloor m-m+1}(n), x_{\left\lfloor \frac{M}{m} \right\rfloor m-m+2}(n), \ldots, x_{\left\lfloor \frac{M}{m} \right\rfloor m}(n)\} \\
&\ldots \\
&x_{\left\lfloor \frac{M}{m} \right\rfloor m}^U(n) = \max\{x_{\left\lfloor \frac{M}{m} \right\rfloor m-m+1}(n), x_{\left\lfloor \frac{M}{m} \right\rfloor m-m+2}(n), \ldots, x_{\left\lfloor \frac{M}{m} \right\rfloor m}(n)\}.
\end{aligned} \tag{2}$$

The envelopes (2) can be aggregated in the following way, for m-step upper envelopes, i.e., $\{x_k^U(n)\}_{k=1}^{k=\left\lfloor \frac{M}{m} \right\rfloor m} = \left[x_1^U(n), x_2^U(n), \ldots, x_{\left\lfloor \frac{M}{m} \right\rfloor m}^U(n) \right]$, $n = 1, 2, \ldots, N$, m succeeding equal values are treated as a single value. So, we can replace each m sequential equal values of the envelope with a single value. The integer part of M divided by m, i.e., $\left\lfloor \frac{M}{m} \right\rfloor$, determines the number of data point in the aggregated envelopes. The aggregated upper envelopes yield a new data series representations formally represented as follows

$$\{y_k(n)\}_{k=1}^{k=\left\lfloor \frac{M}{m} \right\rfloor} = \left[y_1(n), y_2(n), \ldots, y_{\left\lfloor \frac{M}{m} \right\rfloor}(n) \right],\ n = 1, 2, \ldots, N \tag{3}$$

In the case of generating the aggregate lower envelopes, the procedure is similar but the maximum function is replaced by minimum function. The m-step lower approximations (lower envelopes) of a data series (1) is denoted by $\{x_k^L(n)\}_{k=1}^{k=\left\lfloor \frac{M}{m} \right\rfloor m}$ and can be aggregated, i.e., we can replace each m sequential equal values of the lower envelope with a single value, and aggregated envelopes $[y_1(n), y_2(n), \ldots, y_{\left\lfloor \frac{M}{m} \right\rfloor}(n)]$, $n = 1, \ldots, N$, were calculated, and the new representation of the data series with reduced dimensionalities.

Due to introduction of the aggregated envelopes, the dimension of the original data series (1) can be significantly decreased; it means for a given data series (1) of the length M we obtain the reduced length $\left\lfloor \frac{M}{m} \right\rfloor$ of (3), where $m << M$. The aggregated envelopes (both upper and lower) are denoted

as follows $[y_1(n), y_2(n), \ldots, y_{\lfloor \frac{M}{m} \rfloor}(n)]$, $n = 1, \ldots, N$. Dimension of aggregated envelopes is reduced m times and they give another new data series representation thereby the first reduction of dimensionality.

3.2. Generation of the essential attributes

In this section we introduce another representation of data series (3), and at once (1). Each data series (3), for $n = 1, 2, \ldots, N$, describes a point in a $\lfloor \frac{M}{m} \rfloor$-dimension space of real values. In general, we can expect that there is some redundancy of representation dimensionality (Jolliffe, 2002), (Guyon, Gunn, Nikravesh, Zadeh, 2005), and these superfluities can be removed by application of multi-layer feed-forward neural networks (Dreyfus, 2005). There are known applications of auto-associative neural networks (a class of multi-layer feed-forward). The new representation causes additional data compression especially in communication area. Application of auto-associative networks gives lossy compression, it means that lossy compressed data after decompression result similar data to the original, but not exactly the same. Lossy compression over lossless compression is capable of reducing data dimensionality much more. Therefore lossy compression is usually used for audio and image data, but in the current research we will use neural network for compression of data series represented by aggregated envelopes.

Thus, we will use an auto-associative feed-forward neural network with three layers, and two hidden layers. The inputs are described by aggregated envelopes (3) of dimension $\lfloor \frac{M}{m} \rfloor$, while the outputs are decompressed inputs (3), and are described as follows

$$\{\hat{y}_k(n)\}_{k=1}^{k=\lfloor \frac{M}{m} \rfloor} = [\hat{y}_1(n), \hat{y}_2(n), \ldots, \hat{y}_{\lfloor \frac{M}{m} \rfloor}(n)], \text{ for } n = 1, 2, \ldots, N, \tag{4}$$

The aim of such a kind of neural network is to generate features, called here the essential attributes, represented by neurons outputs of the second hidden layer of dimension E, under the assumption that $E << \lfloor \frac{M}{m} \rfloor$. The proposed architecture of the neural network is shown in Fig. 3.

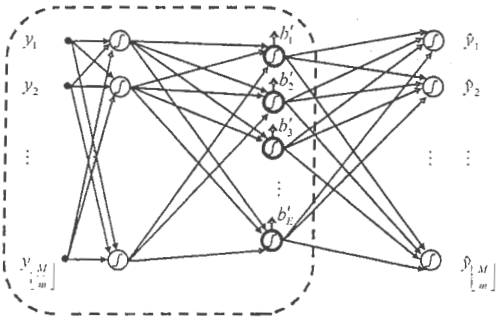

Fig. 3. Neural network generating the essential attributes.

The neural network for generating the essential attributes consists of the following parts:

- the input layer of dimension $\lfloor \frac{M}{m} \rfloor$, for $m << M$, where the inputs represent the aggregated envelopes (upper or lower) $\{y_k(n)\}_{k=1}^{k=\lfloor \frac{M}{m} \rfloor} = [y_1(n), y_2(n), \ldots, y_{\lfloor \frac{M}{m} \rfloor}(n)]$, $n = 1, 2, \ldots, N$,

- the first hidden layer of $\left\lfloor \frac{M}{m} \right\rfloor$ sigmoidal neurons,
- the second hidden layer of E sigmoidal neurons, the outputs of this layer neurons generate signals $\{b_i'(n)\}_{i=1}^{i=E} = \{b_1'(n), b_2'(n), \ldots, b_E'(n)\}$, $n = 1, 2, \ldots, N$, - denoted also by vertical arrows - which are used as the essential attributes, however in the subsequent text we will denote the essential attributes as follows

$$\{b_i(n)\}_{i=1}^{i=E} = 1000\{b_i'(n)\}_{i=1}^{i=E}, \quad n = 1, 2, \ldots, N, \tag{5}$$

because due to using sigmoidal activation function for neurons the second hidden neurons outputs are ranged between -1.0 and +1.0.

The above described three layers: input layer, first hidden layer and second hidden layer all together form a mapping of $\left\lfloor \frac{M}{m} \right\rfloor$ inputs into E essential attributes, it means that this part of the neural network maps each time series represented by an aggregated envelope into a set of essential attributes.

- the output layer of $\left\lfloor \frac{M}{m} \right\rfloor$ sigmoidal neurons denoted by

$$\left\{\hat{y}_k(n)\right\}_{k=1}^{k=\left\lfloor \frac{M}{m} \right\rfloor} = [\hat{y}_1(n), \hat{y}_2(n), \ldots, \hat{y}_{\left\lfloor \frac{M}{m} \right\rfloor}(n)], \text{ for } n = 1, 2, \ldots, N.$$

The whole neural network maps $\left\lfloor \frac{M}{m} \right\rfloor$ input variables into $\left\lfloor \frac{M}{m} \right\rfloor$ output variables, thus the network maps each input into itself. The entire network is necessary to determine weights of connections between neurons of adjacent layers. The weights are obtained during the training process supported by the backpropagation (with modifications) algorithm (Krawczak, 2003a; 2003b).

The following formula expresses the error generated by the network

$$E_r = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{\left\lfloor \frac{M}{m} \right\rfloor}\left(\hat{y}_k(n) - y_k(n)\right)^2 \tag{6}$$

The error (6) is referred as mean square error (MSE)

$$MSE = \frac{1}{N\left\lfloor \frac{M}{m} \right\rfloor}\sum_{n=1}^{N}\sum_{k=1}^{\left\lfloor \frac{M}{m} \right\rfloor}\left(\hat{y}_k(n) - y_k(n)\right)^2 \tag{6a}$$

and just describes the efficiency of compression and decompression of aggregated envelopes data series approximation.

In Fig. 3 the dashed part of the neural network is responsible for compression of data series, and in general only this part is important in our approach of data series dimension reduction. There are known several measures of compression and decompression quality. The most general term for compression measure is *compression ratio* defined as follows (Guyon, Gunn, Nikravesh, Zadeh, 2005)

$$Compression\ ratio = Cr = \frac{Compressed\ Size}{Uncompressed\ Size} \tag{7}$$

In our considered problem developing of the aggregated envelopes causes the ratio is equal to

$$Cr_{aggregation} = \frac{N\left\lfloor \frac{M}{m} \right\rfloor T}{N M T} = \frac{1}{m} \tag{8}$$

where by T we denote number of bits necessary to save separate datum (e.g. 32 or 64 bits), while in the case of the essential attributes the exemplary compression ratio is following

$$Cr_{ess.\ attribute} = \frac{N E T}{N\left\lfloor \frac{M}{m} \right\rfloor T} = \frac{E}{\left\lfloor \frac{M}{m} \right\rfloor}. \tag{9}$$

There is also similar measures e.g. $Space\ Savings = S_s = 1 - \frac{Compressed\ Size}{Uncompressed\ Size}$, or other compressing measures related to speed of data transferring, but there are strictly related to image or audio compressing schemes. However the quality of dimension reduction of data series will be checked by solving illustrative examples in the subsequent part of the paper.

It must be emphasized that the data series representation (1) and the aggregated envelopes representation (3) have a vector form for $n = 1, 2, \ldots, N$, meanwhile the essential attributes representation (5) is obtained a set of attributes. It means that order of the essential attributes does not have any meaning, and it is required to consider the same permutation of attributes for each data series which will be denoted in the following vector form $\{b_i(n)\}_{i=1}^{i=E} = [b_1(n), b_2(n), \ldots, b_E(n)]$, $n = 1, 2, \ldots, N$.

The procedure of generating the essential attributes was described in a general way without distinguished whether there are considered upper or lower aggregated envelopes, and the above consideration about neural networks was done without biases – for simplicity.

In this way we obtained another representation of the original data series (1), and the length E of the new representation indicates additional dimension reduction of data series representation. The idea of the essential attributes obtained from auto-associative neural networks was introduced in earlier papers by Krawczak, Szkatuła (2008) related to mining data series problems.

The essential attributes can be used directly or it is possible to generate a new set of attributes. The new set of attributes $\{c_j(n)\}_{j=1}^{j=K} = \{c_1(n), c_2(n), \ldots, c_K(n)\}$, $n = 1, \ldots, N$, are calculated as rearrangements of differences of the original essential attributes. This way we slightly enlarge dimensionality of the data series representation, but in the same time we provided in some sense the distances between the essential attributes. In particular we have the set of the original essential attributes (5) $\{b_1(n), b_2(n), \ldots, b_E(n)\}$, $n = 1, \ldots, N$, and we generated $K = \binom{E}{2}$ combinations without repetitions of differences $b_i(n) - b_j(n)$, $i, j = 1, 2, \ldots, E$, $i > j$, these combinations create a set of new attributes

$$\begin{aligned}
&\{c_j(n)\}_{j=1}^{j=K} = \\
&= \{b_2(n) - b_1(n),\ b_3(n) - b_2(n);\ b_4(n) - b_3(n),\ \ldots, b_K(n) - b_{K-1}(n), \\
&\qquad b_3(n) - b_1(n),\ b_4(n) - b_2(n),\ \ldots, b_K(n) - b_{K-2}(n), \\
&\qquad\qquad b_4(n) - b_1(n),\ \ldots, b_K(n) - b_{K-3}(n), \\
&\qquad\qquad\qquad \ldots \\
&\qquad\qquad\qquad b_K(n) - b_1(n)\} \\
&= \{c_1(n), c_2(n), \ldots, c_K(n)\}
\end{aligned} \tag{10}$$

for $n = 1, 2, \ldots, N$. The set of attributes $\{c_j(n)\}_{j=1}^{j=K}$ for $n = 1, 2, \ldots, N$ constitutes the new representation of the data series.

3.3. Nominalization of the attributes

In our approach, the real values of the transformed essential attributes (or the essential attributes not transformed) can be replaced by nominal values. In our approach there is no physical interpretation of the essential attributes (or transformed essential attributes), and it is assumed that there are some unknown involved relationships between them, thus the nominalization is done in such a way that the ranges of the all essential attributes values are subdivided into some number of partitions. In the paper, we apply a particular and simple method called equal width interval discretization. In general, the method involves determining the domain of observed values of the attributes and subdividing this interval into equal subintervals. It involves determining the domain of observed values of the all attributes $a_j \in A$, $j = 1,..., K$ and dividing this interval into equal subintervals. The set $V_{a_j} = \{v_{j,1}, v_{j,2},..., v_{j,L_j}\}$ is the domain of the attribute a_j, and L_j denotes the number of subintervals for the j-th attribute. One can construct subinterval boundaries, i.e. cut points, in the following way:

$$\begin{aligned} p_0 &= \min\{V_{a_1}, V_{a_2},...V_{a_K}\} \\ p_i &= p_0 + i \cdot \partial, \quad i = 1,..., P-1 \\ p_P &= \max\{V_{a_1}, V_{a_2},...V_{a_K}\} \end{aligned} \tag{11}$$

where $\partial = \dfrac{\max\{V_{a_1}, V_{a_2},...V_{a_K}\} - \min\{V_{a_1}, V_{a_2},...V_{a_K}\}}{P}$, $P \in N$ is a predefined parameter. Consecutive ranges are labeled by letters of the alphabet, respectively, see Fig. 4.

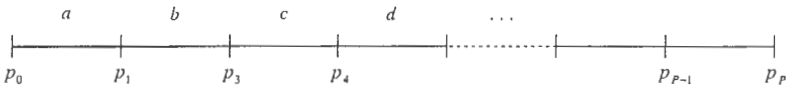

Fig. 4. The nominalization of the attributes.

Such labeling is done for each attribute.

Introducing nominalization of essential attributes can be considered from a point of view of data compression, namely using definition (11) the compression ratio now is rewritten as follows $Cr_{nominalization} = \dfrac{N\,E\,t}{N\,E\,T}$, where t and T denote numbers of bits required to store nominalized attributes and real value essential attributes, respectively.

The set of the new attributes $\{c_j(n)\}_{j=1}^{j=K}$ (or $\{b_j(n)\}_{j=1}^{j=K}$ in the case without transformation) with nominal values now can be denoted by $\{a_j(n)\}_{j=1}^{j=K}$, $n = 1, 2, \ldots, N$, where $K << M$. This way a new representation of data series (1) was developed, the representation is characterized by a set of attributes and values of the attributes are nominal.

4. Illustrative examples

Practical presentation of the proposed approach for the reduction of dimension of data series described by (1) was carried out for the database available at the Irvine University of California (Alcock, Manolopoulos, 1999).

In the next subsections we will discuss the data sets adopted for the calculation and the dimension reduction methodology for data series described in the previous section will be applied to them. The methodology is based on the concept of envelopes, extracting essential attributes, and nominalization of the attributes.

4.1. Description of the data set

The considered database consists of data series synthetically generated by proper equations. Each equation represents a different type of pattern. Each pattern was taken as a time series of 60 data points. The following equations were used to create the data points $z(t)$, where $1 \leq t \leq 60$, for the various patterns:

pattern E: $z(t) = v + rs + kx$ (upward shift)
pattern F: $z(t) = v + rs - kx$ (downward shift)
pattern A: $z(t) = v + rs$, (normal pattern)

where, for each pattern, v is the mean value of the process variable under observation ($v = 80$), s is the standard deviation of the process variable ($s = 5$), r is a random number between -3 and 3, x is the magnitude of the shift (x takes a value between 7.5 and 20), k indicates the shift position in E and F ($k = 0$ before the shift and $k = 1$ at the shift and thereafter).

We considered the following ***learning data*** series where introduced classes correspond to the patterns: 25 time series of Class 1 (pattern E); 25 time series of Class 2 (pattern F); 25 time series of Class 3 (pattern A). Each data series has 60 values, and the whole data series can be described as follows: $\{x_k(n)\}_{k=1}^{k=60} = [x_1(n), x_2(n), ..., x_{60}(n)]$, $n = 1, 2, ..., 75$. All 75 time series after normalization are shown in Fig. 5.

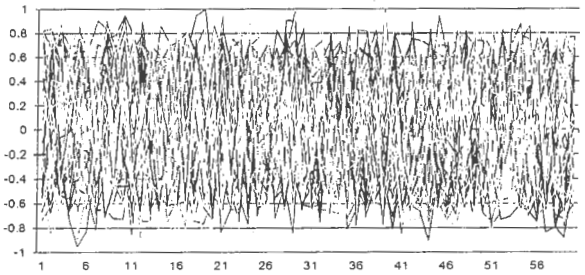

Fig. 5. The normalized learning data series $\{x_k(n)\}_{k=1}^{k=60}$, $n = 1,2,...,75$, belong to the pattern E, F and A.

In Table 1 there are shown exemplary values of selected learning time series of the considered normalized data series.

Table 1
The three selected learning time series.

n	$x_1(n)$	$x_2(n)$	$x_3(n)$	...	$x_{58}(n)$	$x_{59}(n)$	$x_{60}(n)$	Pattern
1	-0.30	-0.60	-0.55	...	0.26	0.12	0.33	E
40	0.26	0.35	0.22	...	-0.73	-0.64	-0.20	F
75	-0.30	0.20	0.22	...	0.51	0.53	0.26	A

We also considered the ***testing data*** series which were different from the learning data: 25 time series of Class 1 (pattern E); 25 time series of Class 2 (pattern F); 25 time series of Class 3 (pattern A), which were used only for testing purposes. Each data series has 60 values, and the whole data series can be described as follows: $\{x_k(n)\}_{k=1}^{k=60} = [x_1(n), x_2(n), ..., x_{60}(n)]$, $n = 76, 77, ..., 150$.

Now, the dimension reduction methodology for data series will be applied step by step.

4.2. Reducing of the data series dimensionality

Our goal is to reduce the dimensionality of the data series under consideration. Dimensionality reduction was started with the creation of m-step upper and lower approximation of a data series. Basic steps of the proposed approach both for upper and lower envelopes are shown below.

4.2.1. Envelopes aggregate approximation.

The value of a step m was determined experimentally as *4*. In order to compare the quality of the approximation of the normalized series of data by the upper and lower approximations, the average deviation of the upper and lower approximation of the all normalized data series was calculated, for different values of m.

Data series representation based on the *4*-step upper approximation of a normalized data series (1) is denoted by $\{x_k^U(n)\}_{k=1}^{k=60}$ and is calculated according to formula (2). Exemplary, the *4*-step upper approximations of a data series number 1 is shown in Fig. 6.

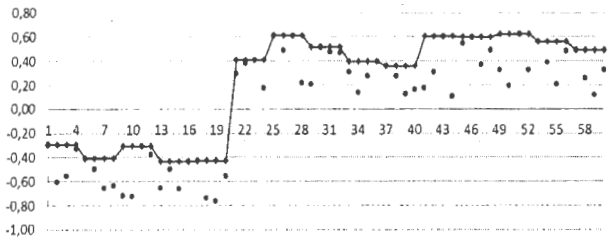

Fig. 6. The *4*-step upper approximations of a data series number 1 for k =1, 2,...,60.

Similarly, data series representation based on the *4*-step lower approximation of a normalized data series (1) is denoted by $\{x_k^L(n)\}_{k=1}^{k=\left\lfloor \frac{M}{m} \right\rfloor m}$. Exemplary the *4*-step lower approximations of a data series number 1 is shown in Fig. 7

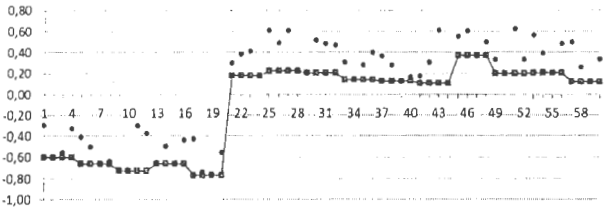

Fig. 7. The *4*-step lower approximations of a data series number 1 for k =1, 2,...,60.

The average deviation of the *4*-step upper and lower approximation of the normalized data series was calculated, i.e., values $\frac{1}{60}\sum_{k=1}^{60}(x_k^U(n)-x_k(n))$ and $\frac{1}{60}\sum_{k=1}^{60}(x_k(n)-x_k^L(n))$, for n-th data series, n = 1, 2, ..., 75, are shown respectively in Fig. 8 a) and 8 b).

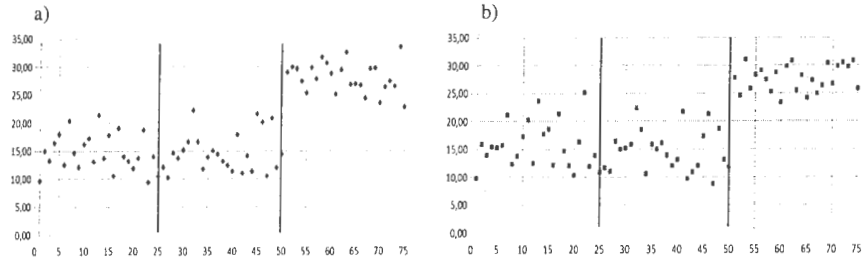

Fig. 8. a) Average deviation of the *4*-step upper approximation of the normalized data series belong to the pattern E (n = 1, 2,...,25), pattern F (n = 26, 27,...,50) and pattern A (n = 51, 52,...,75), b) similarly average deviation of the lower approximation.

As can be seen in Fig. 8 a) and b), similar values of average deviation were obtained for pattern E and pattern F. For data series of pattern A error approximations were slightly greater.

The average deviation of the all data series and lower or upper approximation was calculated as follows $\frac{1}{75}\sum_{n=1}^{75}\frac{1}{60}\sum_{k=1}^{60}(x_k^U(n)-x_k(n))$ and $\frac{1}{75}\sum_{n=1}^{75}\frac{1}{60}\sum_{k=1}^{60}(x_k(n)-x_k^L(n))$. The similar value of average deviation of the all data series was obtained for the upper (takes value 19.13) and lower (takes value 19.33) approximation. So, none of these gives much better approximation and each of them can be used for further calculations. So in the paper two following computational experiments are considered.

- Problem of reducing the dimensionality of data series based on the upper envelopes.
- Problem of reducing the dimensionality of data series based on the lower envelopes.

Calculation results for both the upper and the lower envelopes are given below. For M = 60 and a fixed value m = 4 the number of data point in the aggregated envelopes is 15, according to (3).

For each considered data series the *4*-step upper and lower approximations of the learning data series $[x_1(n), x_2(n),..., x_{60}(n)]$ were calculated and aggregated, and we obtained the new reduced data series representation $[y_1(n), y_2(n),..., y_{15}(n)]$, for $n = 1, 2,..., 75$, see Fig. 9.

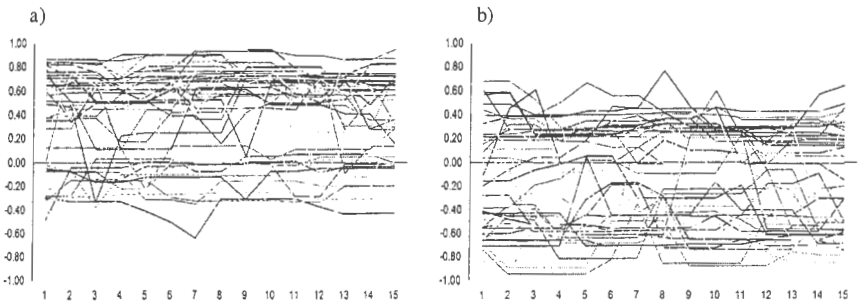

Fig. 9. a) The aggregated 4-step upper envelopes $\{y_k(n)\}_{k=1}^{k=15}$, n = 1,2,...,75, of data series E, F and A, b) similarly aggregated 4-step lower envelopes.

For the data from Table 1 the exemplary aggregated *4*-step upper envelopes look like in Table 2 a) and the aggregated *4*-step lower envelopes look like in Table 2 b).

Table 2

a) The aggregated 4-step upper envelopes.

n	$y_1(n)$	$y_2(n)$	$y_3(n)$	...	$y_{15}(n)$
1	-0.30	-0.33	-0.33	...	-0.43
40	0.35	0.35	0.50	...	0.36
75	0.22	0.74	0.75	...	0.55

b) The aggregated 4-step lower envelopes.

$y_1(n)$	$y_2(n)$	$y_3(n)$	...	$y_{15}(n)$	Pattern
-0.60	-0.60	-0.55	...	-0.74	E
0.21	0.21	0.21	...	0.14	F
-0.30	0.05	0.05	...	-0.64	A

In this way, we obtained the new representation of the data series (1); the new representation is described by either upper or lower envelopes, or by both kinds of the envelopes. According to (8) the compression ratio due to aggregation of envelopes is as follows $Cr_{aggregation} = 0.25$.

4.2.2. Generation of the essential attributes

In order to find the essential attributes of the aggregated envelopes a four layer feedforward neural network (including the input layer) was applied, see Fig. 3, with different numbers of neurons within the second hidden layer. In this particular case the network consists of:

- the input layer of dimension $\left\lfloor \frac{M}{m} \right\rfloor = 15$ representing the aggregated envelopes (upper or lower) $\{y_k(n)\}_{k=1}^{k=15} = [y_1(n), y_2(n), ..., y_{15}(n)]$, $n = 1, 2, ..., 75$,
- the first hidden layer with $\left\lfloor \frac{M}{m} \right\rfloor = 15$ sigmoidal neurons,
- the second hidden layer of E sigmoidal neurons, and the outputs of these neurons $\{b_i'(n)\}_{i=1}^{i=E} = \{b_1'(n), b_2'(n), ..., b_E'(n)\}$, $n = 1, 2, ..., 75$, indicate the essential attributes,
- the output layer of $\left\lfloor \frac{M}{m} \right\rfloor = 15$ sigmoidal neurons generating $\{\hat{y}_k(n)\}_{k=1}^{k=15}$, for $n = 1, 2, ..., 75$.

For designing the required neural network JNNS – a freely available neural network simulator was applied. During the experiment the number of hidden neurons was changed, from 1 up 15, and for each case the network was trained and the learning error described as follows

$$Error = \frac{1}{15} \sum_{n=1}^{75} \sum_{k=1}^{15} (\hat{y}_k(n) - y_k(n))^2$$

was calculated. For adjusting weights we used the backpropagation with momentum algorithm and within each calculation lasting 10000 cycles both required parameters, namely learning parameter and momentum parameter, were selected in such a way to get a stable as well as the lowest value of the learning error. The results of the experiment are shown in Fig. 10 a) where the learning error is drawn vs. the number of applied hidden neurons.

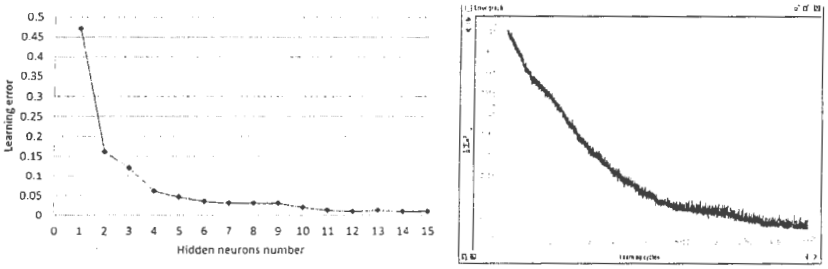

Fig. 10. a) Values of learning error vs. the number of hidden neurons, b) learning error for E=5.

According to the experiment, the number of neurons of the hidden layer was chosen as $E = 5$, see Fig. 10 b), meaning that just five essential attributes are enough to conserve the information about the data series character and the error $Error = 0.05$ corresponds to the absolute value of an average difference between the output and input is equal to 2.5 %. Thus, the used for the further calculation neural network has the architecture shown in Fig. 11.

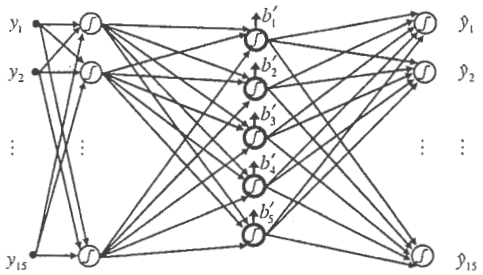

Fig. 11. Neural network generating five essential attributes.

In the next step of our procedure, the outputs of the hidden layer $\{b'_1(n), b'_2(n), \ldots, b'_5(n)\}$ for $n = 1,2,\ldots,75$ were multiplied by 1000. Thus, the original data series can be represented by a set of five essential attributes $\{b_1(n), b_2(n), \ldots, b_5(n)\}$ according to (5).

Due to formula (9) the compression rate of generating essential attributes takes the following value $Cr_{ess.\ attribute} = 1/3$, and together we obtained the compression rate $Cr_{aggregation+ess.attribute} = 1/4 \times 1/3 = 1/12 = 0.08(3)$.

For further consideration, the essential attributes are arranged in a vector form, i.e. one chosen permutation of them must be taken for all investigated data series. Thus, the original learning data series represented by upper envelopers can be represented by a vector of five essentials attributes $[b_1(n), b_2(n), \ldots, b_5(n)]$, see Fig. 12 a), and for upper and lower envelopes see Fig. 12.

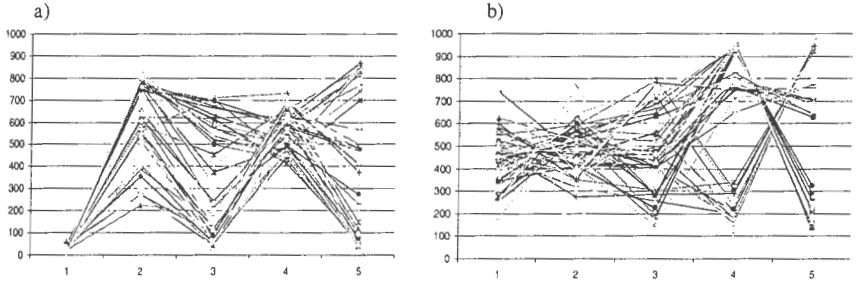

Fig. 12. a) The value of the attributes $\{b_j(n)\}_{j=1}^{j=5}$, $n = 1,2,\ldots,75$, for upper envelopers, b) similarly for lower envelopes.

The exemplary values of the five essential attributes for examples number 1, 40 and 75 with the aggregated *4*-step upper envelopes are shown in Table 3a) and for *4*-step lower envelopes are shown in Table 3b).

Table 3

a) The attributes for the upper envelopes.

n	$b_1(n)$	$b_2(n)$	$b_3(n)$	$b_4(n)$	$b_5(n)$
1	74	400	71	470	834
40	24	761	613	414	63
75	23	283	119	682	558

b) The attribute for the lower envelopes

$b_1(n)$	$b_2(n)$	$b_3(n)$	$b_4(n)$	$b_5(n)$	Pattern
372	454	559	214	946	E
249	713	787	934	228	F
732	387	762	645	865	A

In the paper two following problems are considered: the essential attributes can be used directly, so that the number of attributes is 5; and on the base of the essential attributes new transformed attributes can be generated. The new attributes were calculated as rearrangements of the essential attributes according to (10). Fig. 13 shows new of transformed attributes $\{c_j(n)\}_{j=1}^{j=10}$, $n=1, 2, \ldots, 75$, for a fixed permutation of the attributes for the aggregated *4*-step upper envelopes.

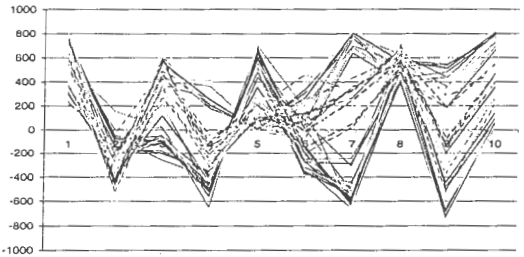

Fig. 13. Plots of the new attributes $\{c_j(n)\}_{j=1}^{j=10}$, $n=1,2,...,75$, for the upper envelopes.

Fig. 14 shows new transformed attributes $\{c_j(n)\}_{j=1}^{j=10}$, $n=1, 2, \ldots, 75$, for a fixed permutation of the attributes for the aggregated *4*-step lower envelopes.

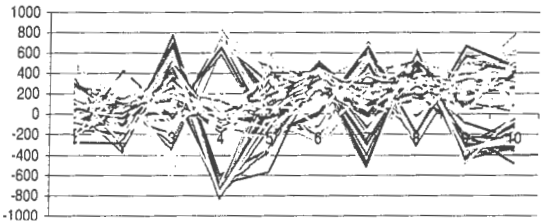

Fig. 14. Plots of the new attributes $\{c_j(n)\}_{j=1}^{j=10}$, $n=1,2,...,75$, for the lower envelopes.

The new attributes $\{c_j(n)\}_{j=1}^{j=10}$, $n=1,2,...,75$ take values from a range [-1000; 1000] and before their nominalization should be fixed in a new range [0; 1000], in order to unify the nominalization process. The following simple formula changes the attributes values ranges: $c_j(n) := \frac{1}{2}c_j(n) + 500$, $j=1,2,...,10$ and $n=1,2,...,75$.

4.2.3. Nominalization of the attributes

Next, the real values of the essential attributes from a range [0, 1000] can be replaced by nominal values. The number of nominal values P corresponding to (11) was determined experimentally as 10. The replacement is done in such a way that the ranges of the all essential attributes are divided into ten partitions. One can construct interval boundaries, i.e. cut points, in the following way (11):

$$p_0 = \min\{V_{a_1}, V_{a_2}, \ldots V_{a_{10}}\} := 0$$
$$p_i = p_0 + i \cdot 100, \; i=1,\ldots,9$$
$$p_{10} = \max\{V_{a_1}, V_{a_2}, \ldots V_{a_{10}}\} := 1000.$$

Hence, the range of all attributes between 0 and 1000 is divided into ten equal sub-ranges. Consecutive sub-ranges are labeled by first ten letters of the alphabet, respectively, as it is shown in Fig. 15. Such labeling is done for each attribute $\{a_j(n)\}_{j=1}^{j=5}$, $n = 1,2,...,75$.

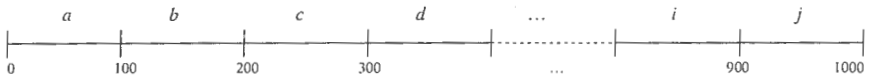

Fig. 15. The nominalization of the attributes

The values of the nominal essential attributes for the aggregated *4*-step upper envelopes are shown in Fig. 16 a); for the aggregated *4*-step lower envelopes are shown in Fig. 16 b).

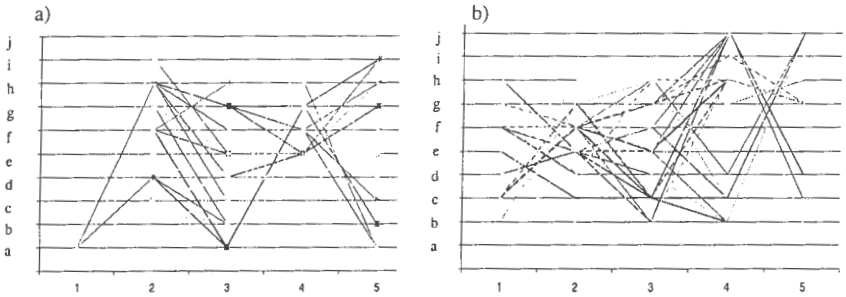

Fig. 16. a) The attributes $\{a_j(n)\}_{j=1}^{j=5}$, $n = 1,2,...,75$, for the aggregated *4*-step upper envelopes, b) similarly for the aggregated *4*-step lower envelopes.

Exemplary values of the essential attributes after the nominalization for the aggregated *4*- step upper and lower envelopes are shown in Table 4 a) and Table 4 b).

Table 4

a) The attributes for upper envelopes

n	$a_1(n)$	$a_2(n)$	$a_3(n)$	$a_4(n)$	$a_5(n)$
1	*a*	*d*	*a*	*e*	*i*
40	*a*	*h*	*g*	*e*	*a*
75	*a*	*c*	*b*	*g*	*f*

b) The attributes for lower envelopes

$a_1(n)$	$a_2(n)$	$a_3(n)$	$a_4(n)$	$a_5(n)$	Pattern
d	*e*	*f*	*c*	*j*	E
e	*e*	*e*	*j*	*b*	F
h	*d*	*h*	*g*	*i*	A

We can notice that in the space of these five attributes there are several examples overlapping, see Fig. 16. It means that some different examples are described by exactly the same values of attributes, i.e. some examples are not distinguishable, as exemplary shown in Table 5.

Table 5

The exemplary values of the attributes for upper envelopes.

n	$a_1(n)$	$a_2(n)$	$a_3(n)$	$a_4(n)$	$a_5(n)$	Pattern
1	*a*	*d*	*a*	*e*	*i*	E
3	*a*	*d*	*a*	*e*	*i*	E
20	*a*	*d*	*a*	*e*	*i*	E
23	*a*	*d*	*a*	*e*	*i*	E
27	*a*	*h*	*g*	*f*	*a*	F
42	*a*	*h*	*g*	*f*	*a*	F
44	*a*	*h*	*g*	*f*	*a*	F
52	*a*	*f*	*b*	*g*	*d*	A
58	*a*	*f*	*b*	*g*	*d*	A

The nominalization of a new set of the real values of all transformed essential attributes $\{c_j(n)\}_{j=1}^{j=10}$, $n=1,2,...,75$ was arranged in a similar way (11). In result we obtained the nominal representation of the data series $\{a_j(n)\}_{j=1}^{j=10}\}$, $n=1,2,...,75$, and values of the attributes take one of the nominal values: *a, b, c, d, e, f, g, h, i, j*. Thus after nominalization each data series is represented by nominal values of ten attributes.

The values of the nominal transformed essential attributes for the aggregated *4*-step upper envelopes are shown in Fig. 17 a), and values of the nominal transformed essential attributes for the aggregated *4*-step lower envelopes are shown in Fig. 17 b).

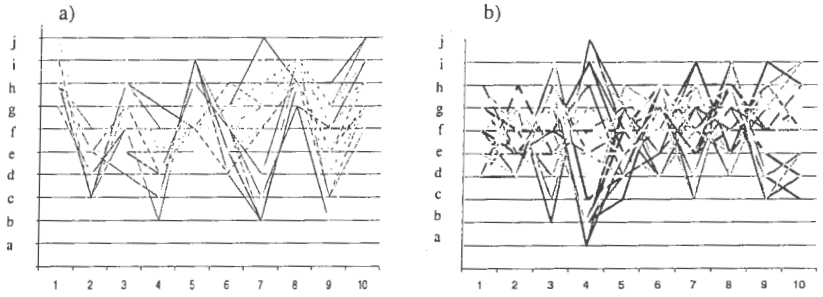

Fig. 17. a) The nominal attributes $\{a_j(n)\}_{j=1}^{j=10}\}$, $n=1,2,...,75$, for the *4*-step upper envelopes, b) similarly for the *4*-step lower envelopes.

Exemplary data series of nominal representations of the transformed attributes for the aggregated *4*-step upper envelopes are shown in Table 6 a); for *4*-step lower envelopes are shown in Table 6 b).

Table 6

a) The transformed attributes for upper envelopes.

n	$a_1(n)$	$a_2(n)$	...	$a_9(n)$	$a_{10}(n)$
1	*g*	*d*	...	*h*	*i*
40	*i*	*e*	...	*b*	*f*
75	*g*	*e*	...	*g*	*h*

b) The transformed attributes for lower envelopes

$a_1(n)$	$a_2(n)$	...	$a_9(n)$	$a_{10}(n)$	Pattern
f	*f*	...	*h*	*h*	E
e	*e*	...	*d*	*d*	F
d	*g*	...	*h*	*f*	A

In the paper the nominal essential attributes $\{a_j(n)\}_{j=1}^{j=5}\}$ and the nominal transformed attributes $\{a_j(n)\}_{j=1}^{j=10}$ are applied for classification problem (Section 5.1); and the transformed attributes $\{a_j(n)\}_{j=1}^{j=10}$ are applied for clustering problem (Section 5.2), for $n=1, 2, ..., 75$.

Assuming 32-bit technology the compression ratio related to bit representation is equal $Cr_{nominalization}=4/32$, and the overall compression rate of generating essential attributes takes the following value $Cr_{SEAA}=1/4\times1/3\times1/8=1/96$.

5. Experimental validation of our dimension reduction

Validation of practical use of the proposed approach was carried out for classification and clustering problems. Calculations were performed to verify whether the proposed methodology of reducing the dimensionality still retains important features of the original data series, which allows to classify or to cluster data series properly.

In order to verify the new data series representation, the database Synthetic Control available at the Irvine University of California was explored. Our approach reduces the dimensionality of an

original time series from 60 data points to 5 nominal values (the attributes $\{a_j(n)\}_{j=1}^{j=5}$) or to 10 nominal values (the transformed attributes $\{a_j(n)\}_{j=1}^{j=10}$) for $n = 1, 2, \ldots, 75$. So data was compressed into 1/12th and 1/6th of its original size, and were treated as learning data. Two the following problems are considered:

- Classification of the aggregate data series (Section 5.1). The aggregation data can be treated as learning data for generation of elementary rules. Classification accuracy is verified by applying the rules for the testing data which did not participate in the generation of the rules.
- Clustering of the aggregate data series (Section 5.2). The aggregate learning data series have been grouped into clusters, but the affiliation of the data series was not used during the computational process, only for testing.

Details of the calculations are presented in the next sections.

5.1. Classification problem

At the beginning of practical verification of SEAA the proposed procedure steps were carried out by performing classification problem. The original data series representation described by real values $\{x_k(n)\}_{k=1}^{k=60} = [x_1(n), x_2(n), \ldots, x_{60}(n)]$, $n = 1, 2, \ldots, 75$, were replaced by another representation described by attributes representation with nominal values $\{a_j(n)\}_{j=1}^{j=K}$, for both $K = 5$ and $K = 10$. The aim is to generate a set of elementary rules for classification of considered data series into one of three classes: Class 1 (i.e., the data series belong to pattern E), Class 2 (i.e., the data series belong to pattern F) and Class 3 (i.e., the data series belong to pattern A).

The attributes were established under both data series representation based on the upper and lower envelopes. These rules could be used for classification of other data series, not classified before, so the rules are verified by using new 75 testing data series $\{x_k(n)\}_{k=1}^{k=60}$, $n = 76, 77, \ldots, 150$, which did not participate in the rules generating process. The selected results are shown below, while details of the measure utilized can be found in paper (Krawczak and Szkatuła, 2011). So, each data series is represented by values of five or ten nominal attributes and now can be treated as learning data for generation of elementary rules of the following form

IF *some conditions are satisfied* THEN *the data series belongs to a proper class.*

In this case, the conditional part of the rules will contain the conjunction of conditions related to the subset of the attributes. The process of generating the decision rules is based on a set of examples under the assumption that for each class the examples have some common properties which distinguish them from another class. The classification accuracy of the rules is understood as percentage of examples correctly classified. In Appendix A the basic elements of the used extraction of decision rules are presented, and details of the inductive learning method used to derive the minimal set of elementary rules can be found in papers by Szkatuła (1995, 2002), Kacprzyk and Szkatuła (1999, 2002, 2005a, 2005b, 2010).

Below, there are results of calculations for the data series representation based on upper and lower envelopes. The calculations take into account both the nominal essential attributes (size of compressed data is 5) as well as the nominal transformed attributes (size of compressed data is 10).

For each case there are shown the generated rules and the consequences of classification accuracy for the learning data and testing.

Case 1: Data series described by nominal essential attributes $\{a_j(n)\}_{j=1}^{j=5}$, for $n = 1, 2, \ldots, 75$.

Each data series, used for learning and for testing) of length 60 was reduced to length 5. All calculations were made for two representations related to the upper and the lower envelopes.

The minimal set of rules for data series representation based on the *4*-step upper envelopes described by five nominal essential attributes was obtained and is shown below.

IF $(a_5 = g \vee h \vee i)$ THEN (Class = 1)

IF $(a_5 = a \vee b)$ THEN (Class = 2)
IF $(a_3 = b \vee c) \vee (a_5 = d \vee e)$ THEN (Class = 3)

These rules correctly classified all of the learning data series represented by nominal attributes of Class 1, Class 2 and Class 3. It is obvious that the generated rules should be verified using another 75 testing data series (25 for each of class) which did not participate in the generation of these elementary rules. In this case the rules correctly classified 73 testing examples, i.e. the classification correctness is about 97.3%.

Described procedure was repeated for the case of data series representation based on the *4*-step lower envelopes described by five nominal essential attributes. The minimal set of generated rules is shown below.

IF $(a_5 = j)$ THEN (Class = 1)
IF $(a_4 = j)$ THEN (Class = 2)
IF $(a_5 = g \vee h) \vee (a_4 = g)$ THEN (Class = 3)

In this case the above rules correctly classified all of the learning data series. For another 75 testing examples (25 of each of class) the rules correctly classified 74 data series, what comprises classification correctness of 98.7 %.

Case 2: Data series described by nominal transformed attributes $\{a_j(n)\}_{j=1}^{j=10}$, for $n = 1, 2, \ldots, 75$.

Again, each data series of length 60 was reduced to length 10. All calculations were made for both data series representation related to the upper and the lower envelopes.

The minimal set of rules for data series representation based on the *4*-step upper envelopes described by ten nominal transformed attributes was obtained and is shown below.

IF $(a_4 = g) \vee (a_7 = i) \vee (a_{10} = j)$ THEN (Class =1)
IF $(a_7 = b \vee c \vee d)$ THEN (Class =2)
IF $(a_7 = e \vee f \vee g \vee h)$ THEN (Class =3)

These rules correctly classified all of the learning data series represented by nominal attributes of to Class 1, Class 2 and Class 3. Then the generated rules were verified using another 75 testing data series (25 for each of class). In this case the rules correctly classified 74 testing examples, i.e. the classification correctness is about 98.7%.

The minimal set of rules for data series representation based on the *4*-step lower envelopes described by ten nominal transformed attributes was obtained and is shown below.

IF $(a_4 = h \vee i \vee j)$ THEN (Class =1)
IF $(a_4 = c \vee b \vee c)$ THEN (Class =2)
IF $(a_4 = d \vee e \vee f \vee g)$ THEN (Class =3)

Then, the rules were verified using another 75 testing data series (25 for each of class), and in this case the rules correctly classified 75 testing examples, i.e. the classification correctness is 100%.

A summary of results obtained for considered classification problems for learning data are shown in Table 7, where the error rate is the ratio of the number of misclassified data series to the total number of data.

Table 7
The results of classification.

Method	Size of the compressed data	Classification accuracy for learning data	The error rate for testing data
SEAA based on 4-step upper envelopes	5	100%	0.027
	10	100%	0.013
SEAA based on 4-step lower envelopes	5	100%	0.013
	10	100%	0.000

The error rates for testing data divided into classes is shown in Table 8, where the error rates are defined as previously.

Table 8
The error rate for testing data divided into classes.

Method	Size of the compressed data	Error rate for Class 1	Class 2	Class 3
SEAA based on 4-step upper envelopes	5	0	0	0.08
	10	0	0	0.04
SEAA based on 4-step lower envelopes	5	0	0	0.04
	10	0	0	0.00

5.2. Clustering problem

Practical verification of the proposed approach was carried out also by performing clustering problem of series described by ten nominal transformed attributes $\{a_j(n)\}_{j=1}^{j=10}$, for $n=1,2,\ldots,75$. The aim is to portion out the set of the considered data series into three, non-empty, disjoint clusters $\{g_1, g_2, g_3\}$, containing all the considered data series, and compare the results with known objects affiliation to classes. It must be emphasized that the affiliation of the grouped data series was not used at all during the clustering process.

The process of clustering of the considered data series was performed according to the clustering algorithm for the nominal attributes which is shortly described in Appendix B. The algorithm belongs to a family of hierarchical clustering algorithms. The clusters descriptions are denoted by $G_{g_1}, G_{g_2}, G_{g_3}$. Each group g can be represented by an ordered collection of values of ten nominal attributes $[a_1, a_2, \ldots, a_{10}]$, i.e., $G_g = < A_{1,t(1,g)}, A_{2,t(2,g)}, \ldots, A_{K,t(K,g)} >$, where $A_{j,t(j,g)} \subseteq \{a, b, c, d, e, f, g, h, i, j\}$, for $j = 1, 2, \ldots, 10$. The details can be found in paper by Krawczak and Szkatuła (in preparation). The applied procedure for the data series described by the upper and lower envelopes is described below.

Case 3: The data series described by nominal transformed attributes $\{a_j(n)\}_{j=1}^{j=10}$, $n = 1, 2, \ldots, 75$.

Again all data series of length 60 was reduced to length 10. Calculations were made for two data series representation related to the upper and the lower envelopes.

Data series representation based on the *4*-step upper envelopes specified by 10 nominal attributes were grouped into three clusters $\{g_1, g_2, g_3\}$, and the clusters descriptions $G_{g_1}, G_{g_2}, G_{g_3}$ are presented in Table 9.

Table 9
The clusters descriptions for data series representation based on the *4*-step upper envelopes.

Description	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
G_{g_1}	*g, h*	*c, d, e*	*g, h, i*	*f, g, h*	*e, f*	*e, f, g*	*i, j*	*g, h, i*	*f, g, h*	*i, j*
G_{g_2}	*h, i, j*	*c, d, e, f*	*d, e, f, g*	*b, c, d*	*g, h, i*	*d, e, f*	*b, c, d*	*g, h, i*	*b, c*	*f*
G_{g_3}	*g, h, i*	*b, c, d, e*	*g, h, i*	*d, e*	*f, g*	*d, e, f, g, h*	*e, f, g, h*	*h, i*	*c, d, e, f, g*	*f, g, h*

The cluster g_1 is represented by $G_{g_1} = <\{g \vee h\}, \{c \vee d \vee e\}, \ldots, \{i \vee j\}>$, while the cluster g_2 is represented by $G_{g_2} = <\{h \vee i \vee j\}, \{c \vee d \vee e \vee f\}, \ldots, \{f\}>$, and the cluster g_3 by $G_{g_3} = <\{g \vee h \vee i\}, \{b \vee c \vee d \vee e\}, \ldots, \{f \vee g \vee h\}>$. The results of clustering shows that all 75 objects were properly grouped according to the objects affiliation during the clustering process, so the clustering efficiency of the nominal attributes data series representation is 100%.

Similarly, data series representation based on the 4-step lower envelopes specification by 10 nominal attributes were grouped into three clusters $\{g_1, g_2, g_3\}$, and the clusters descriptions $G_{g_1}, G_{g_2}, G_{g_3}$ are shown in Table 10.

Table 10
The clusters descriptions for data series representation based on the 4-step lower envelopes.

Description	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
G_{g_1}	*d, e, f*	*f, g, e*	*b, c, d, f, e*	*h, i, j*	*e, f*	*f, d, e*	*i, g, h*	*e, d, f*	*i, h*	*h, i*
G_{g_2}	*e, f, g, h*	*d, e, f*	*h, f*	*b, a, c*	*c, e, g, h*	*h, g*	*d, c*	*g, h, i*	*d, c, e*	*c, d, e*
G_{g_3}	*d, e, f*	*d, g, h, e*	*e, g, h*	*d, e, f, g*	*f, d, e*	*g*	*f, e, g*	*e, f, g, h*	*h, f, g*	*f, g*

In this case, the cluster g_1 is represented by $G_{g_1} = <\{d,e,f\},\{f,g,e\},\{b,c,d,f,e\},...,\{h,i\}>$, while the cluster g_2 is represented by $G_{g_2} = <\{e,f,g,h\},\{d,e,f\},...,\{c,d,e\}>$, and the cluster g_3 by $G_{g_3} = <\{d,e,f\},\{d,g,h,e\},...,\{f,g\}>$.

It is interesting that these three descriptions G_{g_1}, G_{g_2}, G_{g_3} shown in Table 10 can be uniquely distinguished only by only one attribute a_4 or a_{10}. In Fig. 18 in the space of two attributes a_4 and a_8 the data series are marked by shading rhombs.

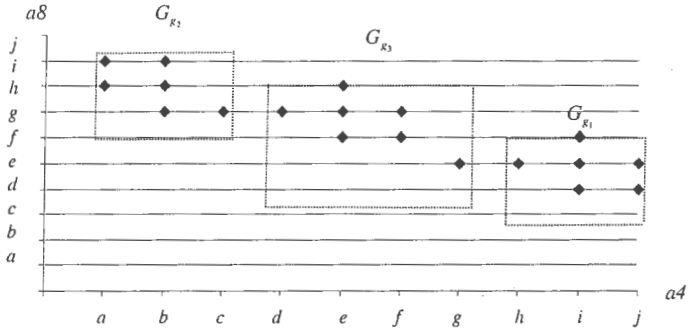

Fig. 18. Three obtained clusters in the space of two attributes a_4 and a_8

It is worth to notice that again the clustering results are perfect and all objects were grouped according to their affiliation, it means the clustering efficiency of the nominal attributes data series representation is 100%. A summary of results for considered clustering problems are shown in Table 10.

Table 10
The results of clustering.

Methods	Size of the compressed data	Size of the alphabet	The error rate
SEAA based on 4-step upper envelopes	10	10	0
SEAA based on 4-step lower envelopes	10	10	0

5.3. Comparative analysis with other method

As it was reported at the beginning of the paper there are many representation techniques that allow us to reduce dimensionality of data series. Simultaneously, little attention has been paid to symbolic representation of data series. It is assumed that different data series representations are introduced in order to reduce their dimension. It seems that combining different technics will give promising results.

One of the most competitive methods in the literature for reducing time series dimensionality with introduced symbolic representation is the symbolic aggregate approximation (SAX). In general this method consists of two main parts, in the first part a time series is approximated by piecewise aggregate approximation (PAA) based on based on piecewise constant approximation (PCA), while in the second part such time series representation is converted into a sequence of symbols, the sequence corresponds to the original time series. SAX method gives reduction of dimensionality not only in the first part but also encoding symbols in bits.

Our introduced method called *symbolic essential attributes approximation* (SEAA) differs considerably from other methods known from the literature, however it is possible to find partial similarities to SAX method. We can say that that SAX consists of two parts, while SEAA of three parts, and it can be said that the first and third part of SEAA are similar to the first and second part of SAX, respectively, see Fig. 1 and Fig. 19.

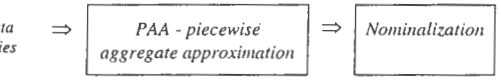

Fig. 19. The approach scheme of SAX - *symbolic aggregate approximation*

Within the first part we generate aggregated envelopes (upper or lower) which are based on piecewise constant approximation obtained for the topmost points, for the upper envelopes, or the lowest points, for the lower envelopes. In some sense envelopes bound the original data series, i.e. the upper envelope from up while the lower envelope from bottom. Such approximations accent changes of the considered original data series. Thus this part of SEAA has some similarities to SAX first part, and in SEAA we can also use different piecewise constant approximation, and it is difficult to say which approximation is better – much depends of particular problem. Therefore, within the first part, both methods are comparable.

In the second part of our approach we generate a set of nominal or symbolic essential attributes, and a chosen permutation of these attributes is kept on. The essential attributes are synthetic and there is no physical interpretation of them but they retain important features of the original data series. In this part further reduction of dimensionality is obtained, and there is not any it has not any counterpart in SAX methodology.

In the third part of SEAA methodology discretization of real value essential attributes are transmuted into symbolic form – in same sense similar as in the second part of SAX approach. In result we can observe certain similarities between these two methods, namely both data series representation have a form of a word over a fixed alphabet.

In SAX, the order of symbols is important and forms a word; a kind of time series. Meanwhile in SEAA, the order of essential attributes in unimportant, but for a fixed permutation a data series representation also constitute a word form. Additionally, in SAX the sequence of symbols is much longer than the value of cardinal number of a set of essential attributes. Similarly, our symbolic value essential attributes can be also encoded in bits giving additional compression rate.

This way we obtain a better compression of ESAA data series representation than SAX can. Therefore the proposed SEAA method is original, little similar to others and only validation procedure can show its efficiency.

A comparison between SEAA and SAX approach for used Synthetic Control Data can be performed for e.g. classification and clustering problems. Results of the calculation for the SAX method are taken from the paper by Lin J., Keogh E., Wei L., Lonardi S. (2007). The error rate was the ratio of the number of misclassified data and the total number of data. SAX obtained an error rate for classification problem of 0.02 for size of the compressed data equal 16 and size of the alphabet equal 10. In the case of application of SEAA approach, for size of the compressed

data equal 5 or 10 and size of the alphabet equal 10, we obtained very similar results for classification as well as for clustering data series, but SEAA due to three level of compression generates gives much greater data compression ratio.

6. Conclusions

In this paper we introduced an approach to reduce dimensions of data series. Our approach differs from algorithms known in the literature. The concept is based on the upper and lower envelopes and their aggregation, and then on essential attributes of the aggregated envelopes. Both representations, related to upper or lower envelopes, allow to obtain high reducing of dimensionality of the original data series. Next, the real values of the transformed essential attributes of data series were converted into nominal values.

A computer experiment was performed for classification and clustering problems. A numerical example shows that even after a very large reduction of dimensionality (as well as reduction of information), the new representation marvelous preserves information about the data series characteristics. Additionally, introducing essential attributes as a set gives possibilities to better compression of data series than in known approaches.

It seems that SEAA method can be elaborated in the future, namely instead of aggregated envelopes any different data series approximations known in literature can be used as inputs of the auto-regressive neural network to generate essential attributes. In general the idea of generating essential attributes as a set of features representing data mining should be explored more deeply.

Appendix A

Extraction of decision rules for classification problem.

In this appendix we present some elements of the inductive learning method used in Section 5.1 to derive the minimal set of elementary rules. Details of the applied approach can be found in papers by Szkatuła (1995, 2002), Kacprzyk and Szkatuła (1999, 2002, 2005a, 2005b, 2010), Szkatuła and Kacprzyk (2005). Therefore, we will recall only such elements of the approach which are necessary to facilitate further considerations.

Let us consider a finite set of examples $e_n \in U$, $n \in \{1,2,...,N\}$. In the case considered in the paper an example is described by K nominal attributes $A = \{a_1,...,a_K\}$ which are referred to as *conditional attributes*. The examples are described in the form of K *elementary conditions* in the following manner:

$$e_n = \bigwedge_{j=1}^{K} (a_j = f(e_n, a_j))$$

where $f(e_n, a_j) = v_{j,t(j,n)}$, and $v_{j,t(j,n)} \in V_{a_j}$, the set $V_{a_j} = \{v_{j,1}, v_{j,2},..., v_{j,L_j}\}$ is the domain of attribute a_j, $j = 1,...,K$, L_j denotes the number of the values of the j-th attribute, $f : U \times A \rightarrow V$ is a function such that $\forall e_n \in U$, $\forall a_j \in A$, $f(e_n, a_j) \in V_{a_j}$. The function $f(e_n, a_j) = v_{j,t(j,n)}$ states that the attribute a_j takes value $v_{j,t(j,n)}$ for the example e_n. The index $t(j,n)$ for $j \in \{1,2,...,K\}$ and $n \in \{1,2,...,N\}$ specifies which value of the j-th attribute occurs in the n-th example.

The conjunction of l elementary conditions, $l \le K$, is written down as: $\bigwedge_{j \in I} s_j = C^I$ where $s_j = (a_j = v_{j,t(j,n)})$, $I \subseteq \{1,...,K\}$, $card(I) = l$. We say that the conjunction C^I *covers* an example e_n if $\forall j \in I$ the condition $f(C^I, a_j) = f(e_n, a_j)$ is satisfied. The set of all the examples described by the conjunction C^I is denoted $[C^I]$.

Suppose that we have additional attribute a_d, called the *decision attribute*, where $\{a_d\} \cap A = \emptyset$ and $V_{a_d} = \{v_{d,1}, v_{d,2},..., v_{d,L_d}\}$ is the domain of the attribute a_d. We can perform the partition of the

entire set of examples into the disjoint classes with respect to the values taken by this attribute. We assume that the number and character of attributes are sufficient for the correct split of examples belonging to different classes. The decision attribute splits the set of examples into the non-empty, disjoint and exhaustive subsets that we call *decision classes*. An example $e \in U$ for which the condition $f(e,a_d)= v_{d,l}$ is satisfied, is called *positive*, while others negative, for the class $U_{v_{d,l}}$, $\forall v_{d,l} \in V_{a_d}$. The sets of the learning examples determined in this manner along with their division into classes, are the starting point in the process of machine learning, which is supposed to lead to the descriptions of the classes considered. In this paper classes are described in the form of rules. In our case, the conditional part of the rules will contain the conjunction of conditions related to the subset of nominal attributes. We say that a rule is *consistent*, if it distinguishes the positive examples from the negative ones. We say that a rule is *minimal* if the removal of any condition from the conditional part of the rule would result in a failure to fulfill the consistency condition. An implication R_k: IF C^{I_k} THEN $(a_d = v_{d,l})$, is called the *k-the elementary rule* for class $U_{v_{d,l}}$, where $C^{I_k} = \bigwedge_{j \in I_k} (a_j = v_{j,t(j,k)})$ is description of example in terms of condition attributes a_j, $j \in I_k$, $I_k \subseteq \{1,...,K\}$ and this example belongs to class $U_{v_{d,l}}$. The index $t(j,k)$ specifies which value of the j-th attribute is used in the k-th rule. The rules, mentioned above, can be formed by applying various algorithms of machine learning. An algorithm used in Section 5.1 can be formulated as follows.

Step 1. Initialize: U^P:= all the positive examples for class $U_{v_{d,l}}$, U^N:= all the negative examples for class $U_{v_{d,l}}$, the initial set of elementary rules R_i $(U_{v_{d,l}})$ is assumed empty, iteration $i = 0$.

Step 2. Iteration $i = i + 1$. We form a modification of the i-th set covering problem on the basis of set examples U^P and U^N. The solution to the i-th set covering problem, provided by the algorithm, determines, in a unique way, the conjunction of conditions forming the elementary rule R_i.

Step 3. Include R_i into the set of rules R_i (U_i), i.e. R_i $(U_{v_{d,l}}) := R_i$ $(U_{v_{d,l}}) \cup R_i$. Eliminate from the set of examples U^P all the examples covered by the rule R_i.

Step 4. If the set of remaining examples U^P is empty, STOP; otherwise, return to Step 2.

In our case, the conditional part of the created rules contains the conjunction of conditions related to the subset of the essential attributes or transformed essential attributes. The rules formed in this manner can be applied to classification of new examples, the ones that have not appeared in the learning process.

Appendix B

Algorithm description for clustering problem.

We consider the problem of clustering the data set described by symbolic attributes. Here we consider the problem of clustering of a set of examples U into non-empty, C disjoints sets $\{g_1, g_2, ..., g_C\}$, where $\bigcup_{i=1}^{C} g_i = U$ and $g_p \cap g_q = \varnothing$, for $p,q \in \{1,...,C\}$, $p \neq q$. It is required that the examples in each cluster are in some sense 'similar', and the examples from different clusters should be 'dissimilar'. The proposed algorithm belongs to a family of hierarchical clustering algorithms. In this method, clusters are built by combining existing clusters, based on their distance (or any other used measure of proximity). The clustering stops when exactly a fixed number of clusters are found. It is assumed that each example must belong to only one cluster. We start with N examples as individual clusters and a pair of clusters described by the lowest value of the clusters' increasing measure is coupled forming a new cluster, and in this way the number of clusters is decreased by one. The details of the measure utilized can be found in paper by Krawczak and Szkatuła (in preparation). Therefore, we will limit the analysis only to these elements of the approach which are necessary to facilitate further considerations.

Let us suppose now that we have a finite set of examples $U = \{ e_n \}$, indexed by n, $n = 1,2,..., N$. The examples are described by K nominal attributes $A = \{a_1,...,a_K\}$ indexed by j. The set $V_{a_j} = \{v_{j,1}, v_{j,2},..., v_{j,L_j}\}$ is the domain of the attribute $a_j \in A$, L_j denotes the number of nominal values of the attribute a_j, $L_j \geq 2$, $j = 1,..., K$. Each example $e_n \in U$ is represented by a K-tuple of sets (i.e. K ordered sets of nominal values) in the following manner $< \{v_{1,t(1,e_n)}\}, \{v_{2,t(2,e_n)}\},..., \{v_{K,t(K,e_n)}\} >$ where $v_{j,t(j,e_n)} \in V_{a_j}$ and $j = 1,..., K$. This notation states that the attribute a_j takes the value $v_{j,t(j,e_n)}$ for the example e_n. The index $t(j,n)$ for $j \in \{1,2,..., K\}$ and $n \in \{1,2,..., N\}$ specifies which value of the attribute a_j is used in the n-th example.

Every non empty group g, $g \subseteq U$, can be represented by an ordered collection of K - sets of values of the attributes describing examples, $G_g = < A_{1,t(1,g)}, A_{2,t(2,g)}, ..., A_{K,t(K,g)} >$, where $A_{j,t(j,g)} \subseteq V_{a_j}$, $card(A_{j,t(j,g)}) \geq 1$ for $j \in \{1,..., K\}$. For instance, for three symbolic attributes $\{a_1, a_2, a_3\}$ and having domains $V_{a_1} = \{a, b, c\}$, $V_{a_2} = \{d, e\}$, $V_{a_3} = \{f, h, n\}$, we can describe an exemplary group g as $< \{a,c\}, \{d\}, \{f,n\} >$ or $< \{a\}, \{d\}, \{n\} >$.

Let us consider two clusters $g_p \subseteq U$ and $g_q \subseteq U$. *Measure of increasing* G_{g_q} *by* G_{g_p} (denoted by $MI(G_{g_p} \mapsto G_{g_q})$) is defined in the following manner:

$$MI(G_{g_p} \mapsto G_{g_q}) = \frac{1}{K} \sum_{j=1}^{K} \frac{card(A_{j,t(j,g_p)} \setminus A_{j,t(j,g_q)})}{card(V_{a_j}) - 1} \cdot \frac{card(V_{a_j} \setminus (A_{j,t(j,g_p)} \cap A_{j,t(j,g_q)})}{card(V_{a_j})}.$$

The measure of increasing is assumed to return a value from [0, 1], where 1 is interpreted as most level for increase, while 0 is the lowest level for increase. The measure is asymmetrical, so this measure should not be considered as the distance between the groups, but the diversity, assuming that one group is a group of the base.

The proposed algorithm is formulated as follows:

Step 1. Each of N example creates one-element cluster in the initial set of clusters $G(U)$, $card(G(U)) = N$, i.e. $G(U) = \{G_{g_1}, G_{g_2},..., G_{g_N}\}$, iteration $k = 0$.

Step 2. Iteration $k = k + 1$. Create a matrix MMI of measures of increasing of the cluster, MMI: $card(G(U)) \times card(G(U))$, where $MMI[p,q] := MI(G_{g_p} \mapsto G_{g_q})$, $p = 1,2,..., card(C(U))$, $q = 1,2,..., card(C(U))$, $p \neq q$.

Step 3. Find two clusters $G_{g_{p^*}}$ and $G_{g_{q^*}}$ that minimize following criterion:

$$MI(G_{g_{p^*}}, G_{g_{q^*}}) := \min_{\substack{\forall p \in \{1,2,..., card(C(U))\} \\ \forall q \in \{1,2,..., card(C(U))\} \\ p \neq q}} MI(G_{g_p} \mapsto G_{g_q}).$$

Step 4. Create a new cluster in the set of clusters $C(U)$, $G_{g_{p^*,q^*}} := G_{g_{p^*}} \oplus G_{g_{q^*}}$. The clusters $G_{g_{p^*}}$ and $G_{g_{q^*}}$ are removed from the set $G(U)$. Thus, $card(G(U)) := card(G(U)) - 1$.

Step 5. If the required number $card(G(U)) = C$ is reached, STOP; otherwise, return to Step 2 and modify the matrix MMI.
The modification of $MMI[p,q]$ relies on removing of the p^*-th and q^*-th rows as well as the p^*-th and q^*-th columns and at the end adding a new row and column. The new row and new column are related to the new cluster $G_{g_{p^*,q^*}}$. The measures $MI(G_{g_{p^*,q^*}} \mapsto G_{g_j})$ for $j = 1,..., card(C(U)) - 1$ and $MI(G_{g_i} \mapsto G_{g_{p^*,q^*}})$ for $i = 1,..., card(G(U)) - 1$ are counted.

In this way, the disjoint set of clusters $G(U) = \{G_{g_1}, G_{g_2},..., G_{g_C}\}$, where $card(G(U))$ - states required number of clusters, is formed.

References

Alcock, Manolopoulos. 1999. http://kdd.ics.uci.edu/databases/synthetic_control/synthetic_control.data.html.

Astrom, K.J. 1969. "On the choince of sampling rates in parametric identification of time series." In *Information Sciences* 1 (3), 273–278.

Azzouzi, M., Nabney, I.T. 1998. "Analysing time series structure with Hidden Markov Models." In *Proceedings of the IEEE Conference on Neural Networks and Signal Processing*, 402–408.

Bagnall, A., Ratanamahatana, C.A., Keogh, E., Lonardi, S., Janacek, G.A. 2006. "Bit level representation for time series data mining with shape based similarity." In *Data Mining and Knowledge Discovery* 13 (1), 11– 40.

Beyer, K.; Goldstein, J.; Ramakrishnan, R.; Shaft, U. 1999. "When is "Nearest Neighbor" Meaningful?". *Proc. 7th International Conference on Database Theory - ICDT'99*, 217–235.

Chan, K.P., Fu, A.C. 1999. "Efficient time series matching by wavelets." In *Proceedings of the 15th IEEE International Conference on Data Engineering*, 126–133.

Chung, F.L., Fu, T.C., Luk, R., Ng, V. 2001. "Flexible time series pattern matching based on perceptually important points." In *International Joint Conference on Artificial Intelligence Workshop on Learning from Temporal and Spatial Data*, 1–7.

Cybenko G. 1989. "Approximations by superpositions of sigmoidal functions." In *Mathematics of Control, Signals, and Systems*, 2 (4), 303-314.

Dreyfus G. 2005. *Neural Networks Methodologyand Applications*. Berlin, Germany, Springer.

Elder, J. IV and Pregibon, D. 1996. "A statistical perspective on knowledge discovery in databases." In *Advances in Knowledge Discovery and Data Mining* (U. M. Fayyad, G. Piatetsky Shapiro, P. Smyth and R. Uthurusamy, eds.) 83-113. AAAI Press, Menlo Park, CA.

Faloutsos C., Ranganathan M., Manolopulos Y. 1994. "Fast subsequence matching in time-series databases." In *SIGMOD Record*, 23, 519-529.

Fu T. C. 2011. "A review on time series data mining." In *Engineering Applications of Arificial Intelligence*, 24, 164-181.

Guyon I., Gunn M., Nikravesh M., Zadeh L. (Eds.), 2005. *Feature extraction foundations and applications.* Berlin, Germany, Springer.

Jolliffe I. T. 2002. *Principal Component Analysis.* Springer.

Johnson S. C. 1967. "Hierarchical Clustering Schemes." In *Psychometrika*, 2, 241-254.

Kacprzyk J., Szkatuła G. 1999. "An inductive learning algorithm with a preanalysis data." In *International Journal of Knowledge - Based Intelligent Engineering Systems*, 3, 135-146.

Kacprzyk J., Szkatuła G. 2002. "An integer programming approach to inductive learning using genetic and greedy algorithms." In L.C. Jain and J.Kacprzyk, eds., *New Learning Paradigms in Soft Computing. Studies in Fuzziness and Soft Computing*. Physica-Verlag Heidelberg, 323-367.

Kacprzyk J., Szkatuła G. 2005a. "A softened formulation of inductive learning and its use for coronary disease data." In *Lecture Notes in Artificial Intelligence*, 3488, 200-209.

Kacprzyk J., Szkatuła G. 2005b. "An inductive learning algorithm with a partial completeness and consistence via a modified set covering problem." In *Lecture Notes in Computer Science*, 3697, 661-666.

Kacprzyk J., Szkatuła G. 2010. "Inductive Learning: A Combinatorial Optimization." In J. Koronacki, Z. W. Ras, S. T. Wierzchoń, J. Kacprzyk, eds., *Advances in Machine Learning. Studies in Computational Intelligence*, 262, Springer.

Keogh, E. 1997. "Fast similarity search in the presence of longitudinal scaling in time series databases." In *Proceedings of the Ninth IEEE International Conference on Tools with Artificial Intelligence*, 578–584.

Keogh E., Chakrabarti K., Pazzani M., Mehrotra S. 2000. "Dimensionality reduction for fast similarity search in large time series databases" In J. Knowl. *Inform. Syst.* 3 (3), 263-286.

Keogh, E., Chakrabarti, K., Mehrotra, S., Pazzani, M. 2001. "Locally adaptive dimensionality reduction for indexing large time series databases." In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, 151–163.

Keogh, E., Chakrabarti K., Pazzani M. 2001. "Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases." In *Proc. of ACM SIGMOD Conference on Management of Data. Santa Barbara. May 21-24*, 151-162.

Keogh, E., Pazzani, M. 2001. "Derivative dynamic time warping." In *Proceedings of the First SIAM International Conference on Data Mining*, Chicago, USA.

Keogh, E., Smyth, P.A. 2001. "Probabilistic approach to fast pattern matching in time series databases." In *Proceedings of the Third ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1997, 24–30.

Krawczak, M. 2003a. *Multilayer Neural Systems and Generalized Net Models*. Ac. Publ. House EXIT, Warsaw.

Krawczak, M. 2003b. "Heuristic dynamic programming - Learning as control problem." In: L. Rutkowski, J. Kacprzyk, eds., *Neural Networks and Soft Computing*. Physica Verlag, Heidelberg, 218-223.

Krawczak, M., Szkatuła G. 2008. "On decision rules application to time series classification." In K. T. Atanassov et al., eds., *Advances in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics*, Ac. Publ. House EXIT.

Krawczak, M., Szkatuła G. 2010a. "Time series envelopes for classification." In *IEEE Intelligent Systems Conference*, London, July 7-9, 2010.

Krawczak, M., Szkatuła G. 2010b. "On time series envelopes for classification problems." In K. T. Atanassov et al., eds., *Developments in Fuzzy Sets, Intuitionistic Fuzzy Sets, Generalized Nets and Related Topics*. Vol II, SRI PAS, Warsaw.

Krawczak M., Szkatuła G. 2010c. "Dimensionality reduction for time series." In *Case studies of the Polish Association of Knowledge*, No. 31, 32-45.

Krawczak M., Szkatuła G. 2011. "A hybrid approach for dimension reduction in classification." In *Control and Cybernetics*, vol. 40, No. 2, 527-552.

Lee, S., Kwon, D., Lee, S. 2003. "Dimensionality reduction for indexing time series based on the minimum distance." In *Journal of Information Science and Engineering* 19, 697–711.

Lin J., Keogh E., Patel P., Lonardi S. 2002. "Finding motifs in time series." In *The 2nd Workshop on Temporal Data Mining, the 8th ACM International Conference on Knowledge Discovery and Data Mining*. Edmonton, Canada, 53-68.

Lin J., Keogh E., Wei L., Lonardi S. 2007. "Experiencing SAX: a Novel Symbolic Representation of Time Series." In *Data Min Knowledge Disc* 2, 15, 107–144.

Maimon O., and Rokach L. (Eds.). 2010. *Data mining and knowledge discovery handbook*.

Matheus, C., Rendell, L. 1989. "Constructive induction on decision trees." In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Mateo, CA: Morgan Kaufmann.

Nanopoulos, A., Alcock, R., Manolopoulos, Y. 2001. "Feature-based Classification of Time-series Data." In *International Journal of Computer Research*, 49-61.

Oja E. 1992. "Principal components, minor components and linear neural networks." In *Neural Networks*, vol.5, 927-935.

Ratanamahatana, C.A., Keogh, E., Bagnall, A.J., Lonardi, S. A. 2005. "Novel bit level time series representation with implications for similarity search and clustering." In *Proceedings of the Ninth Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 771–777.

Rodríguez, J. J., Alonso, C.J. 2004. "Interval and dynamic time warping-based decision trees." In *Proceedings of the 2004 ACM symposium on applied computing* (SAC), 548-552.

Shahabi C., Tian X., Zhao W. 2000. "TSA-tree: A wavelet-based approach to improve the efficiency of multi-level surprise and trend queries." In *Proceedings of the 12th International Conference on Scientific and Statistical Database Management*. Berlin, 55-68.

Shatkay, H., Zdonik, S. 1996. "Approximate queries and representations for large data sequences." In *Proceedings of the 12th IEEE International Conference on Data Engineering*, pp. 536–545.

Smyth, P., Keogh, E. 1997. "Clustering and mode classification of engineering time series data." In *Proceedings of the Thirrd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 24–30.

Szkatula G. 1995. *Machine learning from examples under errors in data*, Ph.D. Thesis, SRI PAS Warsaw, Poland.

Szkatuła G. 2002. "Application of modified covering problem in machine learning." In J. Gutenbaum, ed., *Automatics Control Management*. SRI PAS, Warsaw, 431-445.

Szkatuła G., Kacprzyk J. 2005. "An inductive learning algorithm with a partial completeness and consistency." In: M. Dramiński, P. Grzegorzewski, T. Trojanowski, S. Zadrożny, eds., *Issues in intelligent systems. Models and techniques*. EXIT, Warszawa, 229-246.

Tak-chung Fu. 2011. *A review on time series data mining. Engineering Applications of Artificial Intelligence* 24, 164–181.

Xi, X., Keogh, E., Shelton, C., Wei, L., Ratanamahatana, C.A. 2006. "Fast time series classification using numerosity reduction." In *Proceedings of the 23rd International Conference on Machine Learning*, 1033–1040.

Yang, O., Jia, W., Zhou, P., Meng, X. 1999. "A New approach to transforming time series into symbolic sequences." In *Proceedings of the First Joint Conference between the Biomedical Engineering Society and Engineers in Medicine and Biology*, 974.

Yang, K., Shahabi, C. 2005. "On the stationarity of multivariate time series for correlation-based data analysis." In *Proceedings of the Fifth IEEE International Conference on Data Mining*, 805–808.

Yang Q., Wu X. 2006. 10 "Challenging problems in data mining research." In *International Journal of Information Technology and Decision Making*, 5 (4), 597-604.

Yang, Z., Zhao, G. 1998. "Application of symbolic techniques In detecting determinism In time series." In *Proceedings of the 20th Annual International Conference of the IEEE Engineering In Medicine and Biology Society*, 20 (5), 2670–2673.

Yi B. K., Faloutsos C. 2000. "Fast time sequence indexing for arbitrary L_p norms." In *Proceedings of International Conference on Very Large Data Bases*, Cairo, Egipt.

Yoon, H., Yang, K., Shahabi, C. 2005. "Feature subset selection and feature ranking for multivariate time series." In *IEEE Transactions on Knowledge and Data Engineering, Special Issue on Inteligent Data Preparation* 17 (9), 1186–1198.

Wang B. 2010. "A New Clustering Algorithm on Nominal Data Sets." In *Proceedings of International MultiConference of Engineers and Computer Scientists IMECS 2010*, March 17-19, 2010, Hong Kong.

Wnek, J., and Michalski, R.S. 1994. *Hypothesis-driven Constructive Induction in AQ17-HCI: A Method and Experiments, Machine Learning*, 14, 139-168.

Wu, Y. & Chang, E.Y. 2004. *Distance-function design and fusion for sequence data.* CIKM '04, 324-333.