DATA PREPROCESSING METHODS FOR INTERVAL BASED NEURAL NETWORK PREDICTION
DATU PIRMAPSTRĀDES METODES INTERVĀLU BĀZĒTAI PROGNOZĒŠANAI AR NEIRONU TĪKLIEM

Aleksejs Zorins
Rēzeknes Augstskola
Rēzekne, Atbrīvošanas 115, ITC; e-pasts alex@ru.lv

Abstract. The paper examines a task of forecasting stock prices of Riga Stock exchange by the use of interval value prediction approach, which is carried out by modified Kohonen neural network learning algorithm. The data preprocessing methods are analyzed and implemented here to solve stock prices prediction task. The proposed data preprocessing methods has been experimentally tested with two types of artificial neural networks.

Keywords: artificial neural networks, interval value prediction, Kohonen neural networks, time series transformation methods

Introduction

Neural networks is a very popular direction of artificial intelligence which has found a large number of implementations in such fields as finance, medicine, engineering, geology and physics. Different kinds of network architectures and learning algorithms have been discovered so far. The most popular among them are backpropagation, Elman's, recurrent, counterpropagation networks and Kohonen self-organizing maps [2, 3, 5]. The hybrid networks which combine genetic algorithms, fuzzy logic and neural networks are also being introduced.

One of the tasks, which can be successfully solved by neural networks, is financial forecasting or prediction. Forecasting the behavior of a given system follows two approaches which are trying to predict future values of time series by extracting knowledge from the past. One common approach is to relate the changes in one time series to other phenomena in the economy. Other approach also adopted in this paper, states that stock exchange indexes embody all the knowledge that is necessary to predict future behavior. The experiments presented here are based on the assumption, that the past values of the indexes contain all the knowledge that is necessary to predict future behavior.

Next, there are two possible ways of prediction. Most often a person needs precise or exact values as a prediction result, however, in many cases such a prediction has quite big error and can be replaced with the interval value prediction, when we obtain value (index) change intervals. Varying the length of such intervals, one may obtain required forecasting precision.

The paper examines a task of forecasting stock prices of Riga Stock exchange by the use of interval value prediction approach, which is carried out by modified Kohonen neural network learning algorithm. The data preprocessing methods are analyzed and implemented here to solve stock prices prediction task. The proposed data preprocessing methods has been experimentally tested with two types of artificial neural networks.

Materials and methods

The interval value prediction is based on the partitioning of time series into number of intervals, which may represent factor changes in absolute values or percentage. The simplest case allows using standard windowing method for input value preparation; however, better results are possible with time series transformation methods.

Proper time series transformation can substantially improve performance of the prediction system and reduce learning time. There are different kinds of time series transformation, for instance, Fourier transformation, Wavelets and piece-wise linear approximation [1, 4, 7].
The paper discusses the last one of the methods, which is based on the time series approximation with N straight lines (Fig. 1).

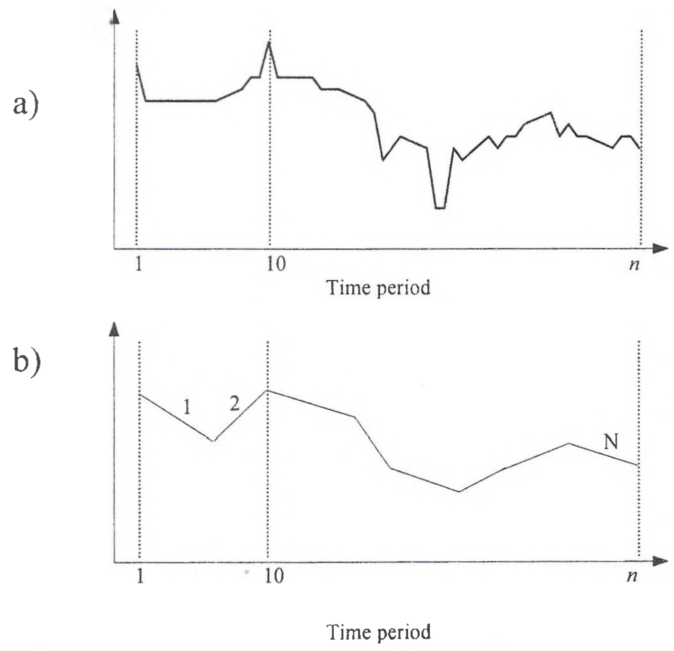


Fig. 1. **Time series (a) and its piece-wise linear approximation**

In most cases $N<n$ (n – number of time series data points), therefore, the time series transformation method allows to process data more efficiently. The time series transformation methods have been initially implemented in data mining, but, as it will be shown later, they may be successfully used as data preparation tool for neural networks.
The goals of the time series transformation (sometimes also called segmentation) may be different [6]:

- time series best representation with fixed number of segments N;
- time series representation in such way, that maximal approximation error for each segment does not exceed initially defined threshold;
- time series representation in such way, that maximum total approximation error for all segments does not exceed initially defined threshold.

There are number of time series transformation (segmentation) algorithms. The experiments presented in this paper deals with three most popular: sliding window, top-down and bottom-up algorithms [4]. The flowchart of the sliding window algorithm is shown in the Figure 2.

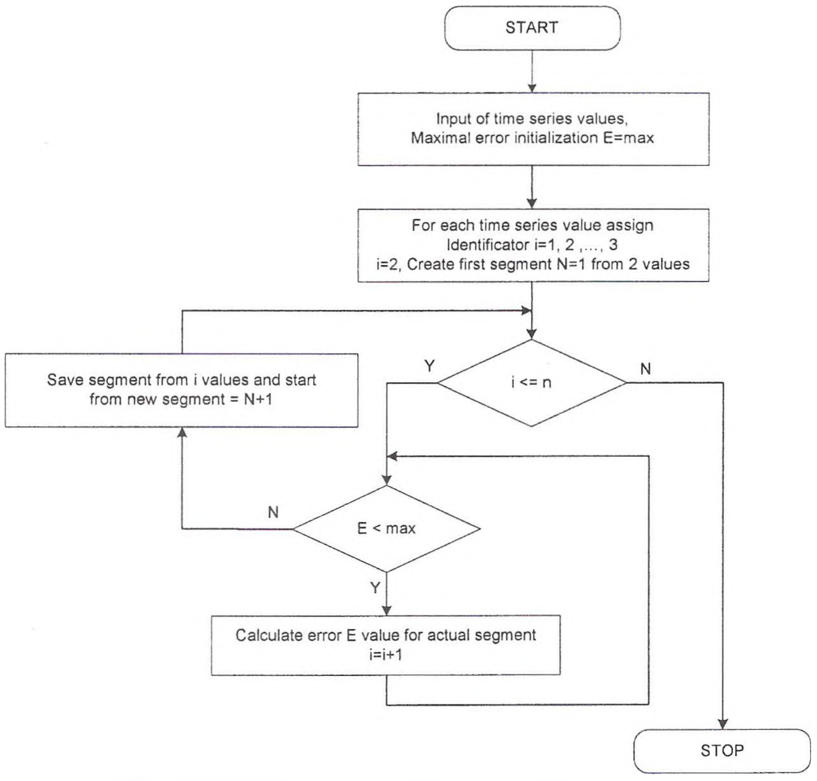

START
Input of time series values, Maximal error initialization E=max
For each time series value assign Identificator i=1, 2 ,..., 3
i=2, Create first segment N=1 from 2 values
i <= n
Y
N
Save segment from i values and start from new segment = N+1
E < max
N
Y
Calculate error E value for actual segment
i=i+1
STOP

Fig.2. **Sliding window algorithm flowchart**

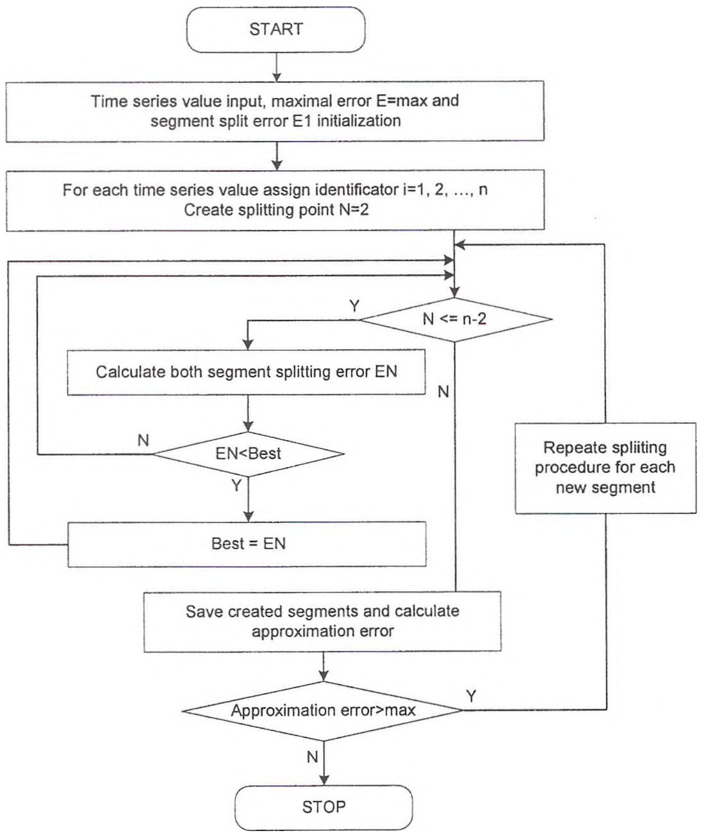

START
Time series value input, maximal error E=max and segment split error E1 initialization
For each time series value assign identificator i=1, 2, ..., n
Create splitting point N=2
N <= n-2
Y
N
Calculate both segment splitting error EN
EN<Best
N
Y
Best = EN
Repeate spliting procedure for each new segment
Save created segments and calculate approximation error
Approximation error>max
Y
N
STOP

Fig.3. **Top-down algorithm flowchart**

Sliding window algorithm starts from the first time series data point, gradually increasing size of a segment and calculating approximation error for each variant of a segment. When the

Emax limit is reached, algorithm creates segment from i-1 time series data points and the process repeats until all time series values are transformed into segments.
The main advantage of this algorithm is its simplicity and possibility to its implementation in on-line regime. There are numbers of possible improvements for this algorithm. For example, it is possible to increase initial step (size of a segment) and use not 1, but k-size step. Experiments show that with k=15 algorithm is 15 times faster and performance (approximation quality) is still good [4].
Figure 3 shows the next time series segmentation algorithm.
The next segmentation algorithm "Top-Down" initially split the time series into 2 equal segments, and then analyze each of them, calculating approximation error. Each of the segments may be split many times, until the desired error threshold is reached. The last time series algorithm, which was used in the experiments, is Bottom-Up. Its flowchart is shown in Figure 4.

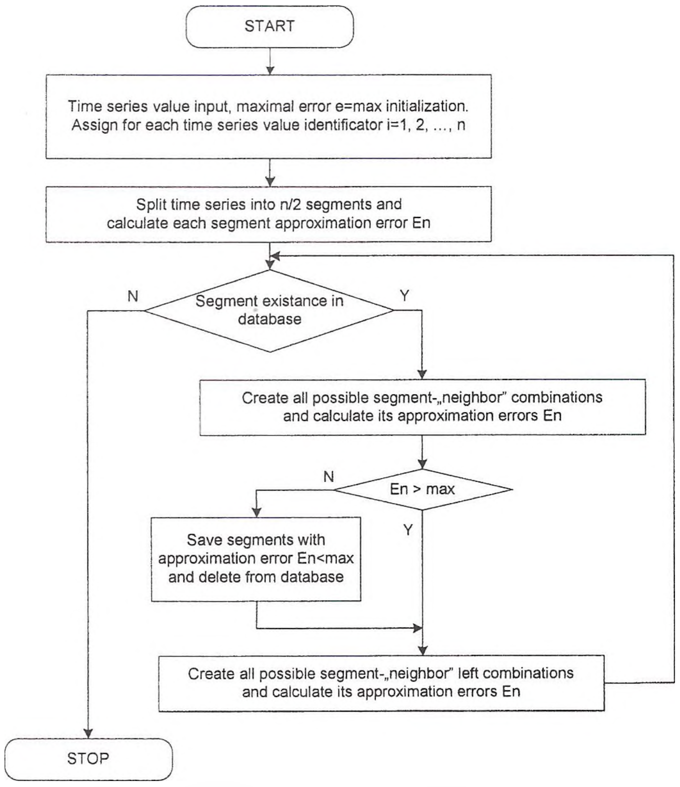

START
Time series value input, maximal error e=max initialization. Assign for each time series value identificator i=1, 2, ..., n
Split time series into n/2 segments and calculate each segment approximation error En
Segment existance in database
N
Y
Create all possible segment-„neighbor" combinations and calculate its approximation errors En
En > max
N
Y
Save segments with approximation error En<max and delete from database
Create all possible segment-„neighbor" left combinations and calculate its approximation errors En
STOP

Fig.4. **Bottom-Up algorithm flowchart**

All three algorithms perform piece-wise linear approximation, which could be calculated in two ways:
- linear interpolation (method simply connects two endpoints of a segment);
- linear regression (segment approximation line is obtained using least square regression).

Linear interpolation requires less calculation and uses less processor time, therefore it is better for on-line segmentation, when time factor is in first place. Linear regression is more time consuming, but allows obtaining better segmentation precision.
In the experiments presented here the time factor is not crucial, and the use of the linear regression instead of the linear interpolation is more appropriate. Regression line is calculated as follows:

$$X_{i+1} = a + bX_i, \tag{1}$$

where
a and b are approximation lines coefficients.

b coefficient is calculated as follows:

$$b = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}, \qquad (2)$$

where

y_i – independent variable;

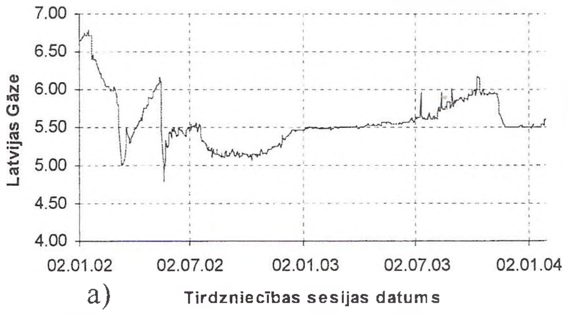
x_i – dependent variable;

n – number of data points.

The next part of the paper shows an implementation of these three algorithms in practical example. All methods are based on the linear regression approximation. Approximation mean square error is used as a benchmark.

Results and discussion

The experiments presented here use the data from Riga Stock Exchange server. The dataset consists of five time series, which represents five Latvian enterprises. Each time series contains 523 data point, therefore allowing testing segmentation algorithm performance. Figures 5 and 6 show the dataset of the task.



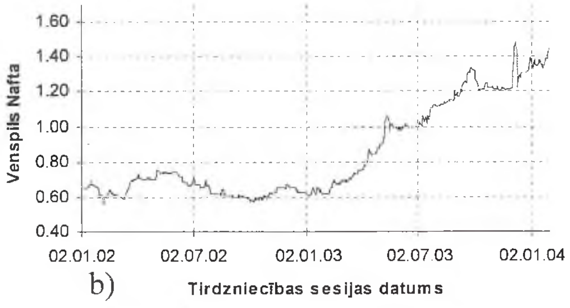
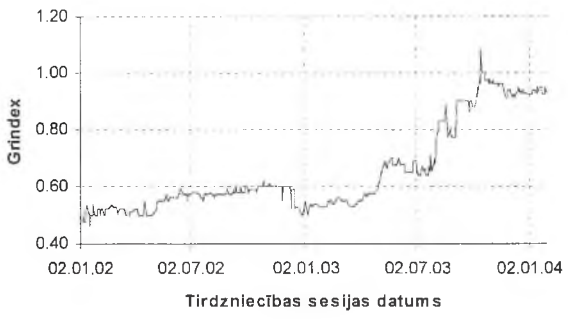
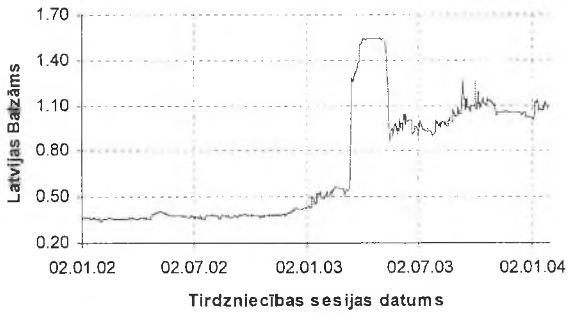


Fig.5. **Time series of Latvian enterprises:** a - Latvijas Gāze, b – Ventspils Nafta





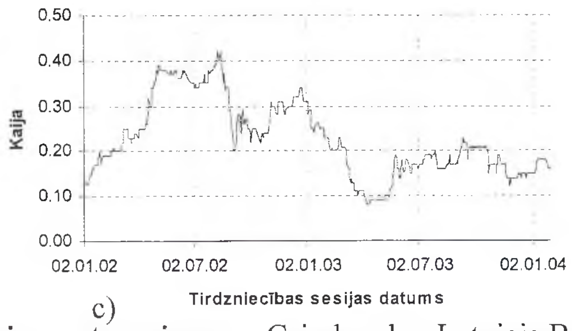


Fig.6. **Time series of Latvian enterprises:** a - Grindex, b – Latviajs Balzāms, c - Kaija

As an additional quality measure for segmentation algorithm the correlation coefficient value will be used. The correlation coefficient in our case shows how well regression line fits data. This coefficient between x and y values is calculated as follows [9]:

$$r = \frac{\dfrac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n}}{\sqrt{\dfrac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n}}\sqrt{\dfrac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n}}}, \qquad (3)$$

where

x_i - actual time series value;

y_i - approximated value;

$\bar{x}$, $\bar{y}$ - mean of the correspondent values;

n - number of data point in time series.

Table 1 summarizes the experimental results on different datasets with different time series transformation algorithms. Additional performance measure here is mean absolute percentage error (MAPE).

Table 1.

Time series transformation results

Time series	*Sliding Window*		*Top-Down*		*Bottom-Up*	
	MAPE	*r*	*MAPE*	*r*	*MAPE*	*r*
Ventspils Nafta (VNFT)	2.42	0.77	2.54	0.72	2.61	0.69
Latvijas Gāze (GAZE)	2.35	0.75	2.28	0.81	2.32	0.73
Kaija (KAIJ)	2.21	0.85	2.31	0.81	2.28	0.82
Grindex (GRDX)	2.39	0.69	2.29	0.72	2.25	0.75
Latvijas Balzāms (BALZ)	2.33	0.81	2.47	0.77	2.31	0.82

Table 1 shows, that none of the methods are better than others in all time series cases. As an additional comparison criteria the number of approximated segments for each time series is used (the less the better).

The best approximation results are possible in the "Kaija" time series with Sliding window algorithm (correlation coefficient r=0.85). The other time series are also transformed rather well.

The time series transformation results may be interpreted in two ways [8]:

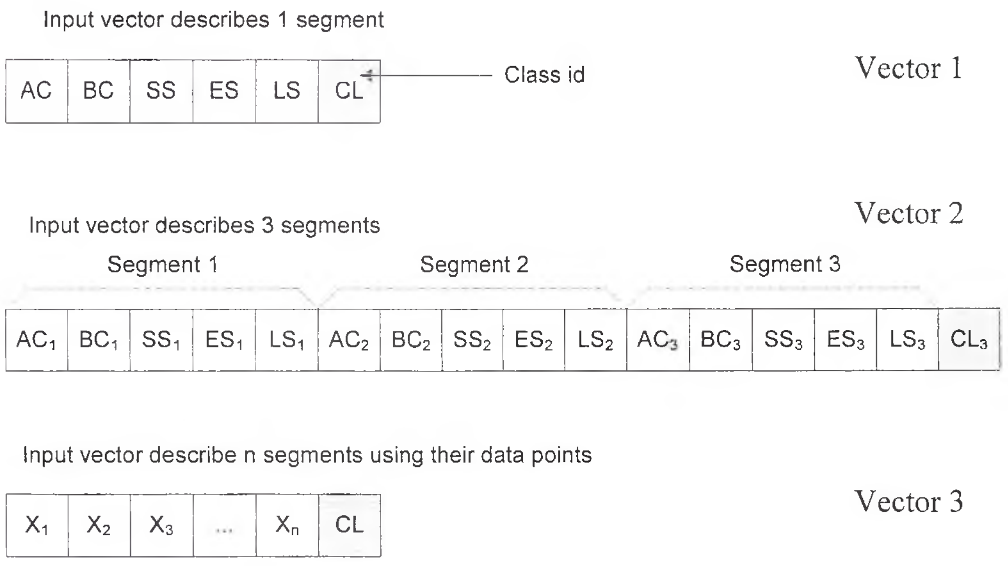
1. Use real time series values from created segments of transformed time series. This approach is similar to windowing approach, but the length of segments may be different.
2. Use as input data from Table 2.

Table 2.

Description of neural network inputs for interval value prediction

Identifier label	*Description*
AC	Linear regression coefficient *a* for each approximated segment
BC	Linear regression coefficient *b* for each approximated segment
LS	Length of a segment (in data points)
SS	Starting point of a segment
ES	End point of a segment
CL	Class identifier

As a result the input vector used in training may be different. Some possible variants are shown in Figure 7.



Input vector describes 1 segment
AC BC SS ES LS CL
Class id
Vector 1
Vector 2
Input vector describes 3 segments
Segment 1
Segment 2
Segment 3
AC₁ BC₁ SS₁ ES₁ LS₁ AC₂ BC₂ SS₂ ES₂ LS₂ AC₃ BC₃ SS₃ ES₃ LS₃ CL₃
Input vector describe n segments using their data points
X₁ X₂ X₃ ... Xₙ CL
Vector 3

Fig. 7. **Neural network input vectors for the experiments**

Let us denote first vector type from the Figure 7 as „Vector 1", the second as „Vector 2" and the third as „Vector 3". The „Vector 1" input type one time series segment. In this case input vector consists of regression equation coefficients, segment starting and ending point, length of a segment and a class identifier.

The experiments show that an input from only one segment does not give enough information for the neural network proper training; therefore it is better to use "Vector 2" kind of input with several segments descriptions and a class identifier.

At this moment we have all information required for the experiments. Table 3 presents Kohonen neural networks performances with different input preparation methods.

Table 3.

Neural networks performances with different input preparation methods

Input preparation method	*Correctly classified objects for the training set (%)*	*Correctly classified objects for the test set (%)*
Vector 1	75	65
Vector 2 (two segment description)	79	72
Vector 2 (three segment description)	**91**	**80**
Vector 2 (four segment description)	85	75
Vector 3	82	63

Conclusions

The proper data preparation methodology is required for its implementation. The time series transformation methods, earlier used in data mining, can also be used in the neural network training experiments for time series prediction.

The table 4 proves that the best results are obtained with Vector 2 data preparation method (when three simultaneously described segments are used in neural network learning). The "common" Vector 3 approach, when the standard windowing method is used, is strongly outperformed by the Vector 2 method. Therefore the data preprocessing method presented here may be till 10% more effective than the standard approach.

References

1. Baestaens D.E., Van den Bergh W.M. (1995). Tracking the Amsterdam Stock Index Using Neural Networks. Neural Networks in Capital Markets, Vol. 5, P. 149-161.
2. Fausett L. (1994). Fundamentals of Neural Networks. Architectures, algorithms and applications, Prentice Hall.
3. Hean-Lee Poh, Jingtao Yao, Teo Jasic (1998). Neural Networks for the Analysis and Forecasting of Advertising and Promotion Impact, International Journal of Intelligent Systems in Accounting, Finance & Management, Vol.7, P. 253-268.
4. Keogh, E., & Pazzani, M. (1998). An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In: Proceedings of the Fourth International Conference of Knowledge Discovery and Data Mining. pp 239-241.
5. Refenes A.N., Zapranis A., Francis G. (1994). Stock Performance Modelling Using Neural Networks, Neural Networks, Vol. 7. No 2. P. 357-388.
6. Rojas R. (1996). Neural networks. A systematic approach, Springer, Berlin.
7. Zirilli J.S. (1997). Financial Predictions using Neural Networks, International Thomson Computer Press, London.
8. Zorin A. (2003). Stock price prediction: Kohonen versus backpropagation, Proceedings of the International Conference "Modelling and Simulation of Business Systems", Vilnius, Lithuania, May 13-14, P. 115-119.
9. Zurada J.M. (1992). Introduction to Artificial Neural Systems, St. Paul: West Publishing Company.