BULETINUL Universităţii Petrol – Gaze din Ploieşti	Vol. LXIII No. 1/2011	185 - 194	Seria Tehnică

Pattern Recognition Via Neural Networks for Adaptive Fault Diagnosis

Konstantin D. DIMITROV*
Inna NACHEVA**

*****Technical University - Sofia, Faculty of Mechanical Engineering,
8, Bul. "Kliment Ohridski", 1756 Sofia, Bulgaria.
e-mail : kosidim @abv.bg

** Technical University - Sofia, Faculty of Automation,
8, Bul. "Kliment Ohridski", 1756 Sofia, Bulgaria.
e-mail : inna.natcheva@gmail.com

Abstract:

The present paper describes a study, that reflects a development of specific Artificial Neural Networks (ANN) training algorithms for Control Charts Pattern (CCPs) recognition. The optimal algorithmic structures are developed and then the most appropriate two kinds of algorithmic structures are studied for Type I and Type II errors. The study was developed during ANN generalization without early stopping (cessation, interruption), as well as for the cases with early stopping. Finally, the best algorithmic structures are proposed. The process modelling, the simulation and the analysis of technical conditions and systems states are developed during fault diagnosis procedures in simulated industrial process and systems condition, that were modeled on an advanced experimental platform, (that was specially designed for the purpose).

Keywords: *Control chart pattern recognition, neural networks, back propagation, generalization, early stopping, fault diagnosis.*

Introduction

The goal of the Pattern Recognition (PR) methods is to classify objects of interest into one of a number of categories or *classes* – [4, 9]. The object of interest are in general called *patterns* and may be electronic wave-forms or signals, actual states of technical systems and /or processes, submitted to analysis and assessment during the Fault Diagnosis (FD) procedures, etc., etc – [6, 9].

If there exists some set of patterns, the individual classes of which are known *a-priori*, then, the PR method could be developed as *a supervised PR* – [9]. The basic structure of a supervised PR-system is shown in **Fig.1**. A subset of labeled patterns (constituting the so-called "training set") is extracted and used for a development of the PR classification algorithm. The remaining patterns (composing the so-called "test set") are used for testing the developed classification algorithm. Since the correct classes of the individual patterns in the test set are also known, the evaluation of the algorithms performance could also be achieved.

If the classes of all available patterns are unknown, and (occasionally) even the number of these classes is also unknown, then the PR method should be developed as *unsupervised PR* (sometimes called also "clustering") – [6, 9].

The clustering PR is based on a selection of classes of patterns, which possess similar properties (sometimes even these properties might not be defined) –[9].

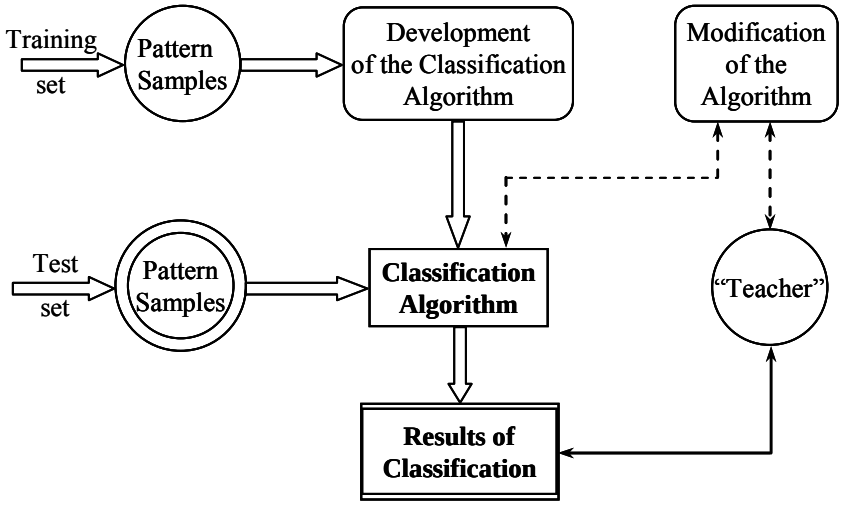

Fig.1. Development of supervised PR

The *Control Charts Pattern (CCP)* Recognition is actually one of the most important tools in statistical process control, designated for identification of the process problems – [6]. Unnatural patterns exhibited by such charts can be associated with certain assignable causes, that affect the analyzed and/or diagnosed processes – [6].

In general, there exist seven basic CCPs, e.g.: *Normal (NOR); Systematic (SYS); Cyclic (CYC); Increasing Trend (IT)*; *Decreasing Trend (DT); Upward Shift (US)* and *Downward Shift (D*S) as shown in **Fig. 2**.

All other types of patterns are either special forms of the basic CCPs or mixed forms of two or more basic CCPs. Only the NOR pattern is indicative for a process continuing to operate under controlled condition. All other CCPs are unnatural and associated with impending problems requiring pre-emptive actions. Enhanced techniques in manufacturing and measurement technology have enabled real-time, rapid and integrated gauging and measurement of process and product quality – [6].

Artificial Neural Systems (ANS), or Artificial Neural Networks (ANN) represent physical cellular systems, which can acquire, store and utilize experimental knowledge – [7, 10]. The knowledge is in the form of stable states or mappings, embedded in the ANN, that can be recalled in response to the presentation of patterns (cues). The ANN can go beyond the digital computers, since they can progressively alter their processing structure in response to the informational data they receive – [1, 5, 8, 10].

The ANN can be trained to *recognize patterns directly* through a typical sample patterns during a training phase. Neural nets may provide required abilities to replace the human operator – [1, 2, 5, 8]. ANN also have the abilities to identify an arbitrary pattern not previously encountered. Back propagation network (BPN) has been widely used to recognize different abnormal patterns of a control chart. BPN is a supervised-learning network and its output value is continuous, usually between [0,1]. It is usually used for detecting, forecasting and classification tasks, and is one of the most commonly used networks – [7, 8, 10].

The present paper describes a study, that reflects a development of specific ANN training algorithms for CCPs recognition. The optimal algorithmic structures are developed and then the

most appropriate two kinds of structures are studied for Type I and Type II errors during ANN generalization for the cases without early stopping (cessation, interruption), as well as for the cases with early stopping. Finally, the best algorithmic structures are proposed.

The present work is based also on the idea, that, with the help of the these types of training algorithmic structures, it would become possible to handle *fault tolerant systems*, in the modality of embedded systems. The idea is to detect errors and to try to correct the failures, that could happen in industrial control by monitoring and fault diagnosis.

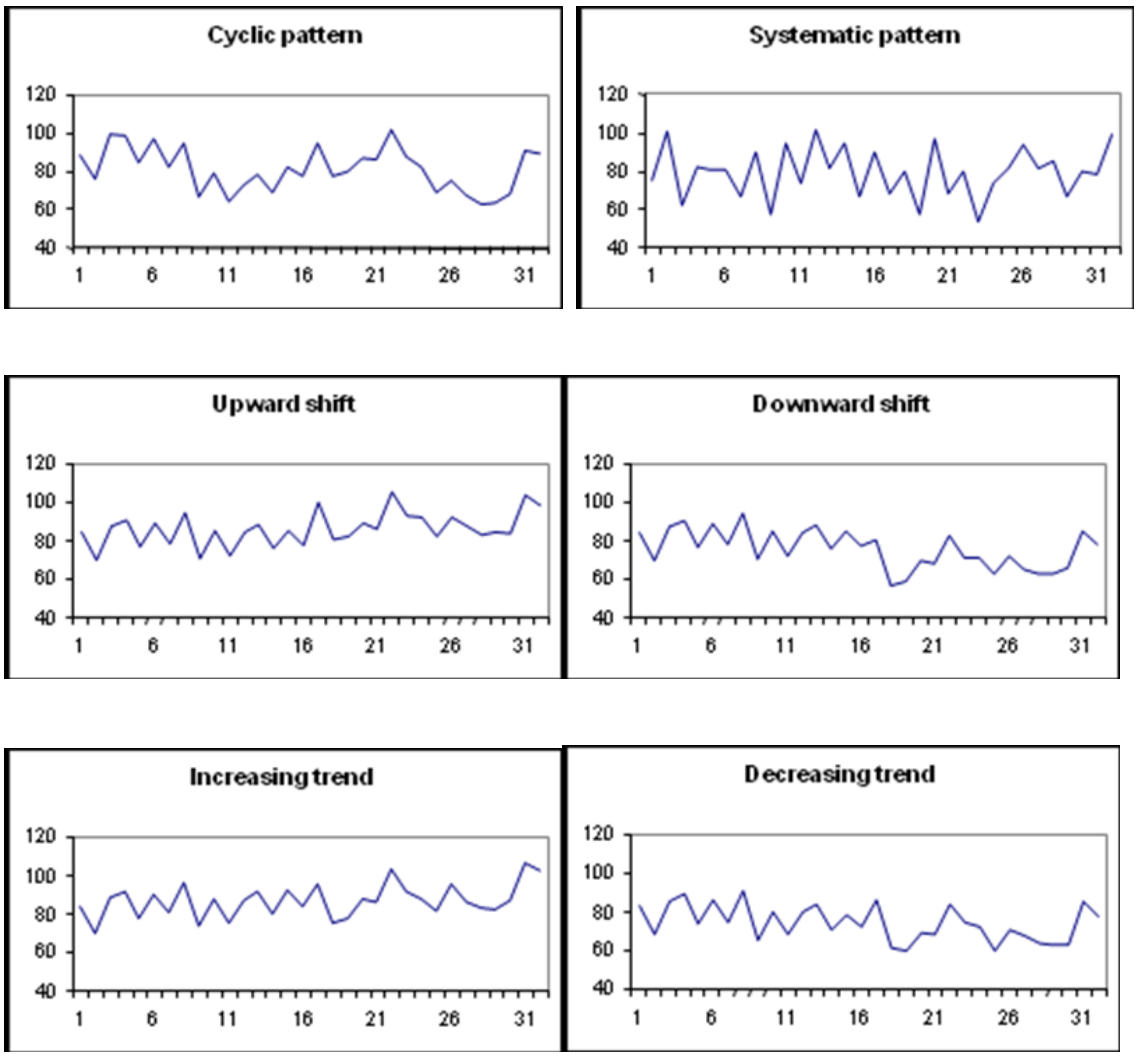

Fig. 2. Abnormal patterns.

The process modeling, the simulation and the analysis of technical conditions and systems states are developed during fault diagnosis procedures in simulated industrial process and systems conditions, that were modeled on an advanced experimental platform – [3], that has been specially designed for the purpose.

Design of the Pattern Recognizer.

Sample patterns.

Sample patterns should be collected from a real manufacturing process. Since, a large number of patterns are required for developing and validating a CCP recognizer, and as those are not often economically available, simulated data, generated *via a specially designed laboratory experimental platform – [3] shall be used.*

Since a large window size can decrease the recognition efficiency by increasing the time required to detect the patterns, an observation window with 32 data points is considered for this particular case. The values of some sets of different parameters for the unnatural patterns are randomly varied in a uniform manner. A set of 3500 (500x7) sample patterns are generated from 500 series of standard normal varieties. It should be noted, that, each set contains equal number of samples for each pattern class. The reason for this is, that, if a particular pattern class is trained many more times, than the others, the Neural Network shall become biased towards that particular pattern.

Some specific equations are developed for the simulation (on an experimental platform) of seven CCPs and respectively applied for the generation of different patterns for the training and testing data sets. The equations are developed as follows:

-Normal Patterns : $y_i = \mu + r_i \sigma$ (1)

- Systematic Patterns : $y_i = \mu + r_i \sigma + d \times (-1)_i$ (2)

- Increasing or Decreasing Trend : $y_i = \mu + r_i \sigma \pm g_i$ (3)

- Upward or Downward shift : $y_i = \mu + r_i \sigma \pm ks$ (4)

- Cyclique Patterns : $yi = \mu + r_i \sigma + a \sin (2\pi_i/T)$ (5)

where “i” is the discrete time point at which the pattern is sampled (i = 1, . . . , 32),
k = 1 , if i ≥ P (point of shift), otherwise k = 0;
r_i - the random value of a standard normal variety at i-th time point;
y_i - the sample value at i-th time point.

ANN-training algorithms

In general, it is very difficult to realize, which training algorithm will be the fastest one for a given problem. The selection and/or the development of a particular ANN-training algorithm depends on many factors, including the complexity of the problem, the number of data points in the training set, the number of weights and biases in the network, the error goal, and this section compares the various training algorithms, etc., etc.

The Backpropagation Algorithm (BPA) uses the gradient of the performance function for the adjustment of the ANNs, stochastic weights, in order to minimize the ANNs performance. During the backpropagation stage, the gradient is determined by performing computations backwards through the entire ANN. There exist many variations of BPA - some of them provide faster convergence, while others have smaller memory requirement.

In the actual study, five particular training algorithms are evaluated and applied for ANNs training: a) The Gradient Descent Algorithms (GDA) - “traindx”;
b) The Resilient Backpropagation Algorithms (RBA) - “trainrp”;
c) The Conjugate Gradient Algorithms (CGA) – “trainscg”;
d) The Quasi-Newton Algorithms (QNA) - “trainbfg”;
e) The Levenberg-Marquardt Algorithms (LMA) – “trainlm”.

The following ***conclusions*** came from the analysis of the ANN training algorithms application:

- The variable learning rate of “traindx” algorithm is usually much slower than the other methods, and has about the same storage requirements as “trainrp” algorithm, but the “traindx” can still be useful for not very complex diagnostic problems.

- The performance of the "trainbfg" algorithm is similar to the one of the "trainlm" algorithm. It does not require as much storage as "trainlm" algorithm, but the required computation increases geometrically with the size of the applied ANN, because the equivalent of a matrix inverse must be computed at each iteration.

- The conjugate gradient algorithms, and the "trainscg" in particular, perform very well over a wide variety of problems, particularly for networks with a large number of weights. The CGA algorithm is almost as fast as the LMA algorithm on function approximation problems (faster for larger networks) and is almost as fast as "trainrp" on *pattern recognition problems*. In many cases, the "trainlm" algorithm is able to provide lower mean-square errors than any other algorithm tested. However, as the number of the weights in the ANN increases, the advantage of "trainlm" algorithm decreases. In addition, the "trainlm" performances are relatively poor on Pattern recognition problems. The storage requirements of "trainlm" algorithm are larger than the other algorithms tested.

- *The "trainrp" is the fastest algorithm on pattern recognition problems*. However, its performance also degrades as the error goal is reduced. The memory requirements for this algorithm are relatively small in comparison to the other algorithms considered.

Based on the experiments, that were developed via MATLAB® , and more specifically its ANN Toolbox [5] , the "traindx" and the "trainrp" algorithms are adopted in this study and applied for training of the ANN, since they provide a reasonably good performance and more consistent results for the problems subjected to the actual study.

Creation of the ANNs structure (topology).

The creation of the ANN structure, (designated for Pattern Recognition) is based on the application of Multilayer Perceptions (MLPs) architecture. The selected structure comprises an input layer, one (or more, depending on the problems complexity) hidden layer(s) and an output layer.

The created MLP-based ANN structure, composed of these layers, as well as its respective weight connections is shown in **Fig.3**. The so-developed pattern recognizer must first be submitted to training and testing, prior to its particular application (i.e., for solving PR-problems).

A *supervised training approach*, is applied for the purpose, and some specific sets of training data (comprising input and target vectors) are presented to the MLP structure.

The learning process is developed via an adjustment of the weight connections between the input and the hidden layers, as well as between the hidden and the output layers.

The weight connections are adjusted in accordance to the specified performances and learning functions. The input nodes dimension is selected to be an equal to the dimension of the observation window, i.e. the number the input nodes is respectively 32 nodes.

The number of the *output nodes* (for the present study) is set to correspond to the number of the *pattern classes*, i.e. to be equal to 7 for this particular case. The labels, shown in Table 1, represent the *targeted values for the recognizers' output nodes*.

The maximal value in each row (selected to be 0.9) identifies the corresponding node and secures the highest output for each pattern (if it is to be considered correctly classified).

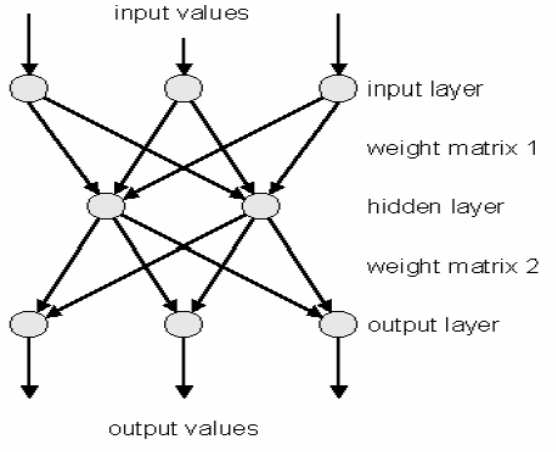

Fig. 3. MLP neural network architecture

Table 1: Targeted recognizer outputs

Pattern class	Recognizer outputs node						
	1	2	3	4	5	6	7
NOR	0,9	0,1	0,1	0,1	0,1	0,1	0,1
SYS	0,1	0,9	0,1	0,1	0,1	0,1	0,1
CYC	0,1	0,1	0,9	0,1	0,1	0,1	0,1
IT	0,1	0,1	0,1	0,9	0,1	0,1	0,1
DT	0,1	0,1	0,1	0,1	0,9	0,1	0,1
US	0,1	0,1	0,1	0,1	0,1	0,9	0,1
DS	0,1	0,1	0,1	0,1	0,1	0,1	0,9

The general rule is, that the ANN size should be as small as possible, for being able to allow an efficient computation. The number of the nodes in the hidden layer is based on the results provided by many experiments conducted via variation of the number of nodes and is selected to be between 11 and 20.

All experiments are coded in MATLAB® , via the application of its NN Toolbox for the selected algorithms "traindx" and "trainrp". The transfer functions used in the calculations are respectively "hyperbolic tangent (tansig)" for the hidden layer and "sigmoid (logsig)" for the output layer. The hyperbolic tangent function transforms the layer inputs to an output interval [−1 , +1] and the sigmoid function transforms respectively the layer inputs to an output interval [0 ,1]. The Coefficient of correlation performances of the developed ANN for the "traindx" algorithm is maximal when the number of nodes in the hidden layer is set to 16. Respectively the value of the coefficient of correlation is maximal when number of the nodes in the hidden layer is set to 17 during the application of the "trainrp" algorithm. The results are shown in **Fig.4**.

The selected ANN architecture is described below:

- ANN details for "traindx" algorithm:
  - topology: 32-16-7 network, with tansig TF and logsig TF in hidden and output layer respectively;
  - training: "traindx" algorithm
- ANN details for "trainrp" algorithm.
  - topology: 32-17-7 network, with tansig TF and logsig TF in hidden and output layer respectively;
  - training: trainrp algorithm.

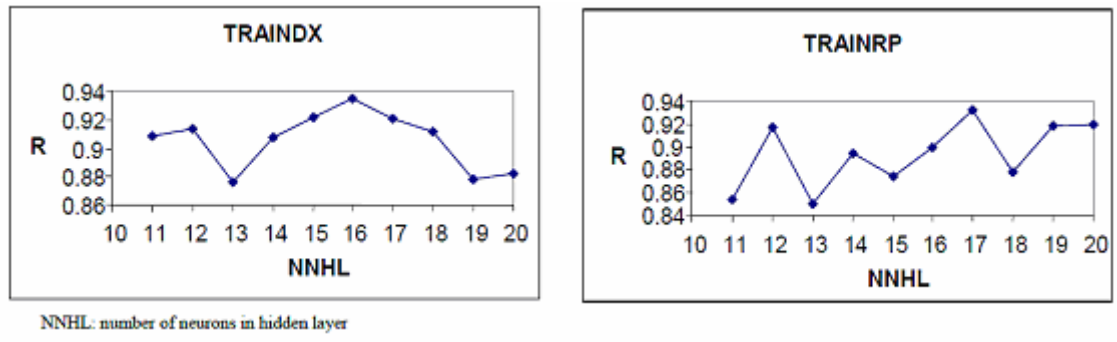

Fig. 4. NNHL VS R

Generalization of the developed ANN.

Improving ANN Generalization

One of the problems, that occur during ANN training is the so-called *over-fitting*. The error on the training set is driven to a very small value, but when new data is presented to the ANN, the error becomes relatively large. The ANN has memorized the training examples, but it has not learned to provide generalization to the new situations. In order to improve the ANN generalization, an early stopping techniques are developed and implemented in the Neural Network Toolbox software.

Early Stopping Techniques.

When developing the so-called "early stopping techniques", the available data are divided into two particular subsets. The first subset is the *training set*, which is used for computing the *gradient and updating the network weights and biases*. The second subset is the *validation set*. The error in the validation set is monitored during the training process. The validation error normally decreases during the initial phase of training, and so does the training set error. However, when the network begins to *over-fit* the data, the error on the validation set typically begins to rise. When the validation error increases for a specified number of iterations ("net. train Param. max_fail = 5"), the training procedure is stopped, and the weights and biases in the minimum of the validation error are returned.

Experimental procedures.

ANN-based Pattern Recognizers are developed using the raw data as the input vector. This section discusses the procedures for the training and the recall (recognition) phases of the Pattern Recognizers. The recognition task was limited to the seven previously mentioned common SPC chart patterns. All the procedures were coded in MATLAB using its ANN toolbox.

Training phase

The overall procedure begins with the generation and presentation of process data to the observation window. All patterns are fully developed during their submission to the recognition window. For the raw data (developed as an the input vector), the pre-processing stage involved their basic transformation into standardized Normal (0, 1) values.

Before their presentation to the ANN for the learning process, the sample data was divided into training (60%), validation (20%) and preliminary testing (20%) sample sets. The so-selected sample sets are then randomized for avoiding some possible bias in the presentation order of the sample patterns to the ANN.

The training procedure was conducted iteratively, and covered ANN *learning*, *validation of in-*

training ANN and preliminary testing. During the learning stage, a training data set (composed of 2100 patterns) is used for updating the network weights and biases. The ANN was then subjected to in-training validation using the validation data set (composed of 700 patterns) for *early stopping*, in order to avoid the over-fitting of the neural network. The error on the validation set would (in general) begin to rise when the network begins to *over-fit* the data.

The training process was stopped when the validation error begun to increase for a specified number of iterations. For the actual study, the maximal number of validation failures was set to be five iterations. The ANN was then submitted to preliminary performance tests using the testing data set (composed of 700 patterns).

The training process was stopped whenever one of the following stopping criteria was satisfied:
- the performance error goal was achieved;
- the maximal affordable number of training epochs was met;
- the maximum number of validation failures was exceeded (i.e., the validation test was fully developed).

Once the training process is entirely completed, the already trained Pattern Recognizer was submitted to an evaluation for acceptance. The recognizer would have to be retrained (via using a totally new data set), if its performances remain poor. Such kind of procedure could be developed to achieve a minimization of the side effects of the poorly trained sets.

Each type of Pattern Recognizer was *replicated* during their exposure to 3 different kinds of training cycles. As a result - 3 different trained recognizers for early stopping were developed and respectively 3 different trained recognizers without early stopping were also developed. The so-developed types of recognizers are labeled as "1.1–1.6", when using "traindx" algorithms and respectively as "2.1–2.6" for "trainrp" algorithms. The results are shown in Table 2. All 6 types of the developed Pattern Recognizers (for both kinds of the training algorithms) possess the same topology and differ only in the applied training data sets.

Recognition Phase (recall) of the trained PR, based on ANN strucutures.

Once developed, the trained pattern recognizers were tested (and more specifically their recall phase) by using 3 different sets, composed of totally new ("fresh") data sets, each one composed of 3500. The results of the recall phase are presented in the Table 2.

The networks were trained *without early stopping*, as well as *with early stopping* via training algorithms "traindx" and "trainrp" and the results are respectively shown in Table 2.

Obtained results.

This section is dedicated to the results, obtained by the performances of Pattern Recognizers, which were trained and tested via two types of algorithms, designated for generalization - respectively *without* early stopping and *with* early stopping.

The particular analysis, performed over of the obtained results showed, that, the Recognition Accuracy, the Coefficient of |Correlation (R) between the actual targets and the predicted targets, as well as the Mean Square Error (MSE) of the algorithms with generalization obtained *with early stopping* ***are higher*** than these provided by the algorithms *without early stopping.*

The training and the recall performances over simulated data bases, (that were randomly generated on a specific laboratory platform - please see [3]), of six types of Recognizers, that were trained respectively with "traindx" and "trainrp" algorithms are shown in **Table 2**. It is obvious, that, the "traindx" algorithms provide better results in two categories analysed.

The overall recognition accuracy for both "traindx" and "trainrp" algorithms is shown in the **Table 3**, as well as in **Fig. 4**. The analysis of the results show, that, the "Type I"-error

performances for both types of training algorithms do not seem to be very adequate. The "Type I" error (please see Table 2.) means a wrong (i.e., a non-adequate) recognition, that *classifies a normal pattern as an abnormal*. The "Type II" error (please see again Table 2.) means also a wrong recognition, that *classifies an abnormal pattern as normal one*. The reason for this, could be explained by the application of highly stochastic and unpredictable structures of the entirely randomly generated data streams (on the specific laboratory platform – [3]), a fact, which made them extremely difficult to be recognized, compared to some real (but also unstable) industrial patterns.

Table 2. Computational results

Traindx with early stopping				Training		Testing	
	R	mse	epoch	Type I	Type II	Type I	Type II
1.1	0,9266	0,0586	156	91,11	99,18	90,4	98,76
1.2	0,9355	0,0580	174	87,77	99,18	89,0	99,22
1.3	0,9263	0,0583	170	91,11	99,34	87,0	99,00
Mean	0,9294	0,0583	166,6	89,99	99,23	88,8	98,99
Range	0,0092	0,0006	18	3,34	0,16	3,4	0,46
Traindx with out early stopping				Training		Testing	
	R	mse	epoch	Type I	Type II	Type I	Type II
1.4	0,8835	0,0613	300	85,55	99,00	88,62	98,10
1.5	0,9129	0,0591	300	87,77	99,34	88,40	98,69
1.6	0,8995	0,0593	300	90,00	98,68	85,60	98,66
Mean	0,8986	0,0599	300	87,77	99,00	87,54	98,48
Range	0,0294	0,0022	0	4,45	0,66	3,02	0,68
Trainrp with early stopping				Training		Testing	
	R	mse	epoch	Type I	Type II	Type I	Type II
2.1	0,9188	0,0578	61	88,88	99,18	86,6	98,5
2.2	0,9097	0,0678	42	82,82	98,33	85,4	98,06
2.3	0,8931	0,0610	40	86,66	98,68	85,4	98,16
Mean	0,9072	0,0625	47,6	86,12	98,73	85,8	98,24
Range	0,0257	0,0091	21	6,06	0,85	1,2	0,44
Trainrp with out early stopping				Training		Testing	
	R	mse	epoch	Type I	Type II	Type I	Type II
2.4	0,8852	0,0661	100	84,3	98,67	89	98,7
2.5	0,8792	0,0649	100	83,33	98,84	86	98,86
2.6	0,8849	0,0686	100	85,04	98,48	85,6	98,33
Mean	0,8831	0,0665	100	84,223	98,66	86,86	98,66
Range	0,006	0,0037	0	1,71	0,36	3,4	0,630

Table 3. Comparison of algorithms

SI no	Recognition accuracy	Traindx	Trainrp
1	With early stopping	94,25583	92,2225
2	Without early stopping	93,20083	92,104417

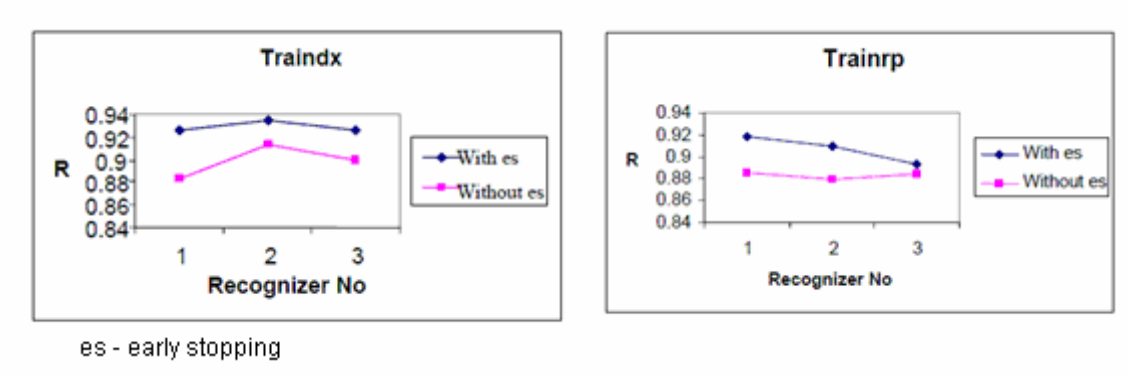

Fig. 5. Comparisons of algorithms performances.

It should be emphasized however, that, the unstable (but obtained during some real industrial processes) data streams have a strong tendency to correlate with the successive data, and as consequence - the structures of the real industrial patterns are more predictable, than the simulated ones, i.e., the recognition of unstable but "industrial" patterns is much easier.

Conclusions:

Specific Artificial Neural Networks (ANN) structures, as well as particular training algorithms for Control Charts Pattern (CCPs) recognition were developed in this study .

The optimal two kinds of algorithmic structures were selected and respectively studied for Type I and Type II errors, during an ANN generalization without early stopping and with early stopping. As a result of this study the best two types of algorithmic structures are proposed.

The experimental procedures were performed over simulated technical conditions of process/systems states, developed during fault diagnosis procedures, modeled on an advanced experimental platform, (that has been specially designed for the purpose)

References

1. Dimitrov, K.D., - *Process Modeling in Industrial Systems via Neural Networks*, XVIII National Scientific and Technical Conference with International Participation - "ADP – 2009", pp 582 - 592.
2. Dimitrov, K.D., - *Some application of Neural Networks for Condition Monitoring in Industrial Systems,* XVIII National Scientific and Technical Conference with International Participation - "ADP – 2009", pp 593 – 598.
3. Dimitrov, K.D., D. I. Nurkov, *Experimental Platform for Process Modeling, Simulation and Analysis of Technical Conditions and Fault Diagnosis,* **"RECENT Journal"**, ISSN 1582-0246, Vol. 11 (2010), N°1 (28), March 2010, Brasov, Romania.
4. Dimitrov,K.D., I.Nacheva, D.Nurkov, - *An automated, modular system for pattern recognition and evaluation of process characteristics in industrial systems,* HCTech – 2008, Revue: Engineering Design, Vol.1., pp. 63-69, Sofia, 2008
5. Demuth, H., Beale, M. - *Neural Network Toolbox User's Guide,* Natick, MA: Math Works, 1998,
6. Hassan, A., etc., - *Improved SPC chart pattern recognition using statistical features*, International Journal of Production Research, 41(7), 1587–1603, 2003.
7. Rich, E., Knight, K. - *Artificial intelligence*, Second Edition, Ed. Mc Graw - Hill, p. 476-478, 1993.
8. Smith, M. - *Neural networks for statistical modeling*. Van Norstrand Reinhold, New York, 2005.
9. Therrien, C.W. - *Decision Estimation and Classification – An Introduction to Pattern Recognition and Related Topics,* John Willey and Sons, 1999.
10. Zurada, J.M. - *Artificial Neural Systems*, West Publishing Company, 1999.