Artificial Neural Network Approach for Fault Recognition in a Wastewater Treatment Process

Mihaela Miron, Laurențiu Frangu, Sergiu Caraman, Laurențiu Luca
Faculty of Automation, Computers, Electrical Engineering and Electronics
"Dunărea de Jos" University of Galați
Galați, Romania
(Mihaela.Miron, Laurentiu.Frangu, Sergiu.Caraman)@ugal.ro

***Abstract*— The paper deals with fault detection and recognition for WWTP (Wastewater Treatment Plant). The chosen classifier is a feed-forward neural network. Its input is a high-size vector of measured variables, rather than a small-size compressed feature vector. The output of the network points to the recognized fault class. The test was performed on a simulated WWTP, disturbed by 6 different types of faults (sensors and actuators). The results of the test proved a good ability of the neural network to recognize the faults, in 97.2% of the analysed cases.**

Keywords—*wastewater treatment process; neural networks; fault isolation; fault recognition; actuator fault; sensor fault; process diagnosis;*

I. INTRODUCTION

This paper deals with the fault detection and recognition, in a wastewater treatment process (WWTP). These processes are complicated and have a highly nonlinear model. According to [1] they operate in harsh conditions and present high risks for the environment, if operated improperly. Faults of the sensors and actuators can occur, so a detection and recognition system is necessary, in order to avoid hazardous consequences. Supplementary, deviations of the biological population dynamics can occur, which are treated also as faults [2], [3]. Previous work of our group dealt with fault detection for WWTPs, but no recognition was attempted [4], [5]. The problem of fault detection and recognition itself is very complex and implies: estimation of the non-measurable variables [4], [6], [7], use of residuals for detection and recognition [5], [6], high dimensional data handling [3], [8], pattern recognition automata, including the neural network approach of the recognition [8]. According to classic works and the field of pattern recognition [9], different groups of techniques can be used for classification: parametric or non-parametric, deterministic or statistic. Usually, these techniques apply to small size feature vectors, as various compression methods are used for reducing the vector size. For instance, papers such as [3], [6], deals with the fault detection and recognition, for WWTP. They use various techniques for fault classification ranging for model based to data driven techniques. The size of the classified vector is rather small.

Among the deterministic methods, the neural networks seem to be a promising approach, since they do not need prior knowledge about the model of the plant. Although, high size input vectors do not represent advantage itself, neural networks can coup with this aspect of the data processing. In this paper, we investigate the ability of the neural network to classify vectors of measurable variables in a WWTP, without making use of compression techniques. Instead, full size vectors of measurable variables recorded over a limited horizon constitute the input of the neural network, as in [8], [9]. The classes are the faults to be recognized and the output of the network is a vector pointing to the recognized fault class. The particular example we use for testing the proposed classification method concerns sensors and actuator faults.

The paper structure is as follows: the second section presents the mathematical model of a wastewater treatment process; the third section describes the neural network approach in order to recognize the faults; the fourth section presents the results obtained and the last section is dedicated to the conclusions.

II. THE MATHEMATICAL MODEL

A wastewater treatment process, presented in [10], [4] was chosen to illustrate the fault recognition technique proposed in this paper. The model is described by the following equations:

$$\frac{dX}{dt} = (\mu(t) - \mu_s(t))X(t) - D(t)(1+r)X(t) + rD(t)X_r(t) \quad (1)$$

$$\frac{dS}{dt} = -\frac{\mu(t)-\mu_s(t)}{Y}X(t) - D(t)(1+r)S(t) + D(t)S_{in} \quad (2)$$

$$\frac{dDO}{dt} = -\frac{(1-Y)(\mu(t)-\mu_s(t))X(t)}{Y}\cdot 10^3 - D(t)(1+r)DO(t) + 60\alpha W(t)\big(DO_{sat} - DO(t)\big) + D(t)DO_{in} \quad (3)$$

$$\frac{dX_r}{dt} = D_s(t)(1+r)X(t) - D_s(t)(\beta+r)X_r(t) - -0.5D_s(t)(1+\beta)X_r(t) \quad (4)$$

$$\mu(t) = \mu_{max}\frac{S(t)}{K_S+S(t)}\cdot\frac{DO(t)}{K_{DO}+DO(t)} \quad (5)$$

$$D = \frac{F_{in}}{V}; D_s = \frac{D\cdot V}{V_s} \quad (6)$$

where $X(t)$ – biomass concentration, $S(t)$ – substrate concentration, $DO(t)$ – dissolved oxygen concentration, $X_r(t)$ – recirculated biomass concentration, $\mu(t)$ – specific growth rate, μ_{max} – maximum specific growth rate, $D(t)$ – dilution rate, $W(t)$ – aeration rate, r – recirculating rate, S_{in} – influent substrate concentration, DO_{in} – influent dissolved oxygen concentration, DO_{sat} – saturation value of dissolved oxygen, Y – yield coefficient, K_s – saturation constant of the substrate, K_{DO} – saturation constant of dissolved oxygen, α – oxygen transfer rate, β – the rate of the sludge in excess, F_{in} – influent flow, V – bioreactor volume, D_s – the dilution rate of the sludge, V_s – sludge volume. Table 1 describes the parameters and initial conditions of the mathematical model:

TABLE I. PARAMETERS AND INITIAL CONDITION OF THE PROCESS MODEL

μ_{max}	0.11 [h^{-1}]
D(0)	0.025 [h^{-1}]
W(0)	5 [$L \cdot min^{-1}$]
r	1
$S_{in}(0)$	0.8 [$g \cdot L^{-1}$]
DO_{in}	2 [$mg \cdot L^{-1}$]
DO_{sat}	8 [$mg \cdot L^{-1}$]
Y	0.67
K_s	0.18 [$g \cdot L^{-1}$]
K_{DO}	0.2 [$g \cdot L^{-1}$]
α	0.0033 [L^{-1}]
β	0.2
V	35 [L]
V_s	6 [L]
X(0)	0.5 [$g \cdot L^{-1}$]
S(0)	0.8 [$g \cdot L^{-1}$]
DO(0)	2 [$mg \cdot L^{-1}$]
$X_r(0)$	0 [$g \cdot L^{-1}$]

III. THE NEURAL NETWORK APPROACH

A. The method

This section describes a fault recognition method using a feed-forward neural network to isolate the faults which can occur in WWTPs. According to [1], a wastewater treatment plant is a complex process "where sensors and equipment are operated at harsh conditions, and there are often long time delays in variables response to disturbances". Therefore, the types of faults analyzed in this process are net and partial faults, which can occur at the level of measurement and control equipment (sensors and actuator components). Pattern recognition techniques use the observed symptoms and compare them to a set of known symptoms for each type of fault by searching the best fit. In this case, the observed symptom is a vector, containing the values of the measurable variables of the process, over a horizon of N sample moments. The samples of the variables go back from the current moment k to the sampling moment k-N+1. The output of the recognition automata is also a vector, called fault vector. Its size is equal to the number of considered faults, plus one for the normal behavior. The fault vector contains values of 0 or 1 (1 - symptom observed in that particular type of fault, 0 – that particular type of fault is not present). This means that the fault vector points to the class of the observed vector. The recorded pattern vectors corresponding to known faults form the training set. Each observed vector of the training set, corresponding to a fault or to normal behavior, is associated to the class it belongs. In order to obtain a good recognition rate, the training set should contain vectors observed in all considered faults and in normal behavior, throughout all functioning regimes. The pattern recognition automata use various techniques: deterministic or statistic, parametric or non-parametric. They make use of the pattern vector (the symptom) and yield the fault vector, i.e. the result of the classification. The neural networks can also act as classifiers – this is the chosen solution in our work. In the field of pattern recognition different types of neural networks can be used. According to [8], the most common used architecture is the feed-forward network, like in multilayer perceptron and Radial-Basis Function (RBF) networks. Also, for data clustering and feature mapping, the Self-Organizing Map (SOM) and the Kohonen networks are popular. In our approach, a multilayer neural network with feed-forward architecture was chosen to act as a classifier. This network has the role of recognizing the net and partial faults that occur in WWTP. The fault recognition scheme, containing the neural classifier, is presented in Fig. 1. The inputs of the neural network are: the measurable input variables $u(k)$ and the measurable output variables $y(k)$, including their history over the last N samples. The output is the result of the recognition, i.e. the previously mentioned fault vector.

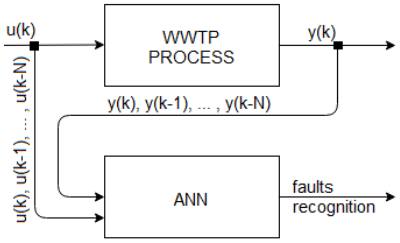

Fig. 1. Fault recognition scheme

B. Design of the neural network

For the system modeled in section 2, the measurable variables are: 5 inputs (D, W, r, b, S_{in}), and 4 outputs (X, S, DO, X_r). The value of dissolved oxygen concentration in the influent is considered constant ($DO_{in} = ct$). Their history over the last 10 samples is collected to form the observed vector (the pattern to be recognized). In all, the size of the observed vector is 90, so the neural network receives 90 inputs. Six types of faults were considered for testing this method. Accordingly, the number of classes to be recognized is 7 and the output of the network contains 7 logic variables. Only one out of 7 should have the value 1, meaning the observed vector was classified to that particular type of fault or to the normal behavior. The form of the fault vector can be:

$[1\ 0\ 0\ 0\ 0\ 0\ 0]^T \in$ Class 1 (normal operation);

$[0\ 1\ 0\ 0\ 0\ 0\ 0]^T \in$ Class 2 (fault of the recirculation pump);

$[0\ 0\ 1\ 0\ 0\ 0\ 0]^T \in$ Class 3 (fault of the supplying pump);

$[0\ 0\ 0\ 1\ 0\ 0\ 0]^T \in$ Class 4 (fault of the excess sludge pump);

$[0\ 0\ 0\ 0\ 1\ 0\ 0]^T \in$ Class 5 (fault of the biomass sensor);

$[0\ 0\ 0\ 0\ 0\ 1\ 0]^T \in$ Class 6 (fault of the dissolved oxygen sensor);

$[0\ 0\ 0\ 0\ 0\ 0\ 1]^T \in$ Class 7 (partial fault of the supplying pump);

The structure of the neural network is represented in Fig. 2, as drawn by Matlab (the environment used for simulation). It contains:

- 1 output layer (7 neurons: 1 class represents the normal operation state and 6 classes correspond to the 6 types of considered faults). The activation function is log-sigmoid, as it has to provide a logic type output;

- 1 hidden layer with 10 neurons. The activation function is also log-sigmoid, in order to behave nonlinearly, as the process does;

- 1 input layer with 90 neurons: 5 input variables (D, W, r, b, S_{in}) and 4 output variables (X, S, DO, X_r), also taking into account their history over the last $N = 10$ samples.

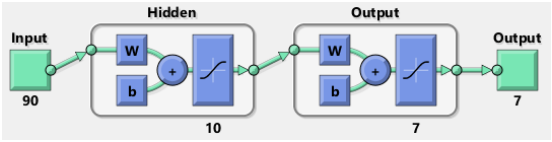

Fig. 2. The structure of the pattern recognition network

The Matlab function used to create the neural network is:

$$net = patternnet(hiddenLayerSize) \qquad (7)$$

where net – the network name, $patternnet$ – the function for creating the pattern recognition network, $hiddenLayerSize$ – numbers of neurons in the hidden layer (set to 10).

C. Training the neural network

Training concerns more steps: collecting the training set and dividing this set in subsets, learning the parameters, validation of the learning process, evaluation of the performance. The training process is represented in Fig. 3.

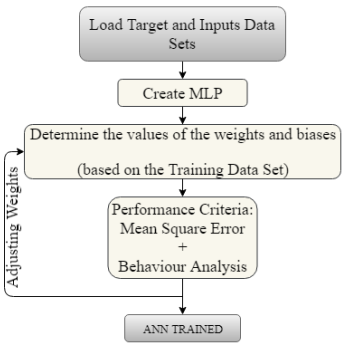

Fig. 3. Artificial neural network training steps

The training set consists of 10889 examples, representing all 7 classes. The examples were collected through simulation of the model described in section 2. Almost 40% of the examples represent the normal behavior, while the other 60% almost uniformly represent the 6 types of faults. For simulating the faults, some parameters of the model (1) – (6) were altered. For the training purpose, two matrices were formed with these examples: the input data set and the output set. The input data set is a 90 x 10889 matrix, containing the observed vectors that will act as network inputs. The output data set (or target set) is a 7 x 10889 matrix, containing the fault vectors associated with each example and pointing to the class where the corresponding input vector belongs. The form of the fault vectors is chosen as they should be yielded by the network.

As the learning step is followed by the validation of the parameters and the evaluation of the classification performance, the input data matrix is randomly divided into three subsets, corresponding to these steps. Of course, the output data set is correspondingly divided into three subsets, of the same size. The three subsets are:

- *learning data set*: used to learn the neural network parameters. At this stage the weights of connections between neurons are determined;

- *validation data set*: analyze the behavior of the neural network during the learning algorithm. The performance analysis that is obtained on the validation data determines whether the neural training process continues or not;

- *testing data set*: determine the performance level of the neural network as a classifier.

The division of the input data set is performed by the Matlab function:

$$net.divideFcn = 'dividerand' \qquad (8)$$

In this case, 70% of the total number of examples (from the input data set) are allocated for the learning, 15% for validation and 15% for testing. That means 7623 for learning, 1633 for validation and 1633 for testing. Table 2 shows the distribution of the examples of the 7 classes into the three data subsets. The first class includes examples describing the normal state of the process, and the other 6 classes contain examples corresponding to the 6 types of faults considered in the following order: net-faults (recirculation pump, supplying pump, excess sludge pump, sensor of the biomass concentration, sensor of the dissolved oxygen concentration) and a partial fault (supplying pump operating at 25% of capacity).

TABLE II. RANDOM DISTRIBUTION OF THE EXAMPLES IN DIFERRENT CLASSES INTO LEARNING, VALIDATION AND TESTING DATA SET

Data Sets	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Total
Input Data Set	4289	1091	1100	1100	1100	1100	1109	10889 (100%)
Learning Data Set	3019	753	760	763	771	779	778	7623 (70%)
Validation Data Set	637	165	170	163	161	154	183	1633 (15%)
Testing Data Set	633	173	170	174	168	167	148	1633 (15%)

The performance function chosen for the learning process is the mean square error (MSE):

$$net.performFcn = 'mse' \quad (9)$$

The learning algorithm is backpropagation, based on a conjugate gradient-minimizing method (*trainscg* – neural transfer function in Matlab) also known as *Scaled Conjugate Gradient Backpropagation*:

$$net.trainFcn = 'trainscg' \quad (10)$$

According to [3], $trainscg$ "is a network training function that updates weight and bias values according to the scaled conjugate gradient method". The actual train is performed by $train$ function form Matlab:

$$[net, tr] = train(net, inputs, outputs) \quad (11)$$

where $train$ – trains the network with $trainscg$, $inputs$ – input data set, $outputs$ - target data set.

The learning and validation processes are repeated until the recognition performance is good enough or the performance stopped improving. As an example, Fig. 4 contains the results of a training process, as diplayed by Matlab. The training process stopped at iteration 336 since the MSE value didn't improve for six consecutive epochs (Fig. 5). Therefore, at the epoch 330, where the validation error value is approximately 0.004, it is assumed that the neural network training process is completed.

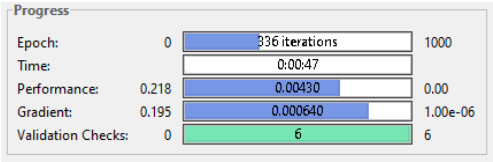

Fig. 4. The training parameters values of the pattern recognition neural network

The MSE error shows the performance of the classifier in recognising all three data subsets (blue - learning, green - validation and red - testing). Fig. 5 shows that during the first 5 epochs the MSE decreasing rate is high and at the following epochs it is decreasing slower. The confusion matrix following the learning step is displayed in Fig. 6.

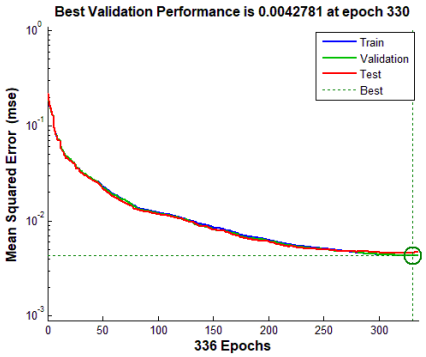

Fig. 5. The evolution of MSE error during training

IV. Results and Discussions

Following the steps of the training process, the ability to correctly classify the examples in the training set should be displayed. In this section, it will be expressed through the confusion matrix and the "Receiver Operating Characteristics" curves. The confusion matrix synthetically presents the right and wrong classification decisions, for all classes, while the ROC curves present the ratio between the false and right decisions, when the network classified a fault. The value "1" in the output vector of the neural network, on the position corresponding to class *j*, means that the input vector was classified to the class *j*. This result can be right or wrong. Accordingly, the evaluation of the accuracy counts one of the following results:

- *True Positive (TP)* - an example belonging to the class *j* (*true*) is recognized by the network as belonging to class *j* (*positive*);

- *False Positive (FP)* - an example belonging to another class than *j* (*false*) is classified by the network as belonging to class *j* (*positive*).

In a similar manner, when the network outputs a "0" on the position of class *j*, the following results can be counted:

- *True Negative* (*TN*) - an example that doesn't belong to class *j (true)* isn't recognized by the network as belonging to this class (*negative*);

- *False Negative (FN)* - an example that does belong to class *j* (negative) isn't classified to this class (*negative*).

The recognition rates are calculated as follows:

- **TP** (true positive rate):

$$r_{TP} = true\ positive/total\ recognized\ positive \quad (12)$$

- **FP** (false positive rate):

$$r_{FP} = false\ positive/total\ recognized\ positive \quad (13)$$

- **TN** (true negative rate):

$$r_{TN} = true\ negative/total\ recognized\ negative \quad (14)$$

- **FN** (false negative rate):

$$r_{FN} = false\ negative/total\ recognized\ negative \quad (15)$$

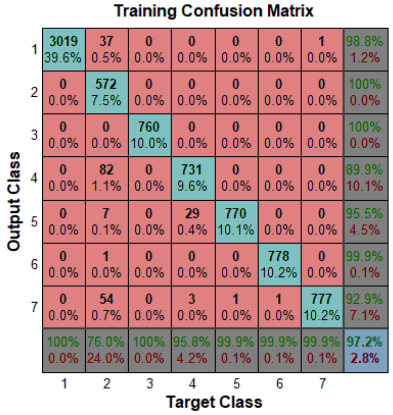

Fig. 6. The confusion matrix resulted after learning step

TABLE 3 RECOGNITION RATES FOR THE LEARNING STEP

Class	No. of examples	TP [%]	FP [%]	TN [%]	FN [%]
1	3019	98.75	1.25	100	0
2	753	100	0	97.43	2.57
3	760	100	0	100	0
4	763	89.91	10.09	99.53	0.47
5	771	95.53	4.47	99.98	0.02
6	779	99.87	0.13	99.98	0.02
7	778	92.94	7.06	99.98	0.02

The rate of correct recognition of all the 7 classes is 97.2% (7407 examples out of the 7623 included in the training set were correctly recognized) and the incorrect recognition rate is 2.8% (216 examples have been incorrectly recognized). The above recognition rates show the performance of the neural network in recognizing the pattern of each class. Table 3 shows that the best recognition rates (in case of training process) are obtained for class 3 and class 6, having very good true recognition rates (class 3: $TP = 100\%$, $TN = 100\%$; class 6: $TP = 99.87\%$, $TN = 99.98\%$) and very low values for false recognition rates (class 3: $FP = 0.13\%$, $FN = 0.02\%$; class 6: $FP = 0\%, FN = 0.05\%$).

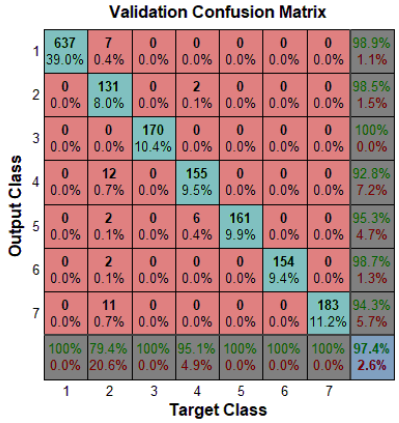

Fig. 7. The confusion matrix resulted after validating the neural network

The result of the validation stage can be observed in the confusion matrix of Fig. 7. The correct recognition rate of all the 7 classes is 97.4% (1591 out of the 1633 examples included in the training set were correctly recognized) and the incorrect recognition rate is 2.6% (42 out of 1633 examples were incorrectly classified).

TABLE 4 RECOGNITION RATES AFTER THE NEURAL NETWORK VALIDATION

Class	No. of examples	TP [%]	FP [%]	TN [%]	FN [%]
1	637	98.91	1.09	100	0
2	165	98.49	1.51	97.73	2.27
3	170	100	0	100	0
4	163	92.81	7.19	99.45	0.55
5	161	95.26	4.74	100	0
6	154	98.71	1.29	100	0
7	183	94.32	5.68	100	0

Table 4 shows that the best recognition rates are obtained for class 1 and class 3. They exhibit high recognition rates (class 1: $TP = 98.91\%$, $TN = 100\%$; class 3: $TP = 100\%, TN = 100\%$) and low values for false recognition rates (class 1: $FP = 1.09\%$, $FN = 0\%$; class 3: $FP = 0\%, FN = 0\%$). After the neural network is tested, the results can be observed in the confusion matrix of Fig. 8. The correct recognition rate of all the 7 classes is 97.4% (1590 examples out of the 1633 included in the training set were correctly recognized) and the incorrect recognition rate is 2.6% (43 examples out of 1633 were recognized incorrectly)

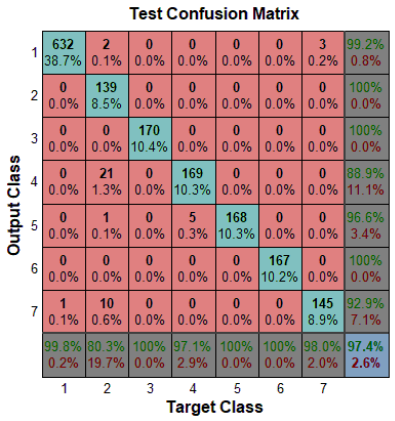

Fig. 8. The confusion matrix resulted after testing the neural network

The recognition rates are shown in Table 5. The best recognition rates are obtained for class 3 and class 6 (class 3: $TP = 100\%$, $TN = 100\%$; class 6: $TP = 100\%$, $TN = 100\%$). They avoided false recognition, so the false recognition rates are class 3: $FP = 0\%, FN = 0\%$ and class 6: $FP = 0\%, FN = 0\%$.

TABLE 5 RECOGNITION RATES AFTER TESTING THE NEURAL NETWORK

Class	No. of examples	TP [%]	FP [%]	TN [%]	FN [%]
1	633	99.21	0.79	99.89	0.01
2	173	100	0	97.72	2.28
3	170	100	0	100	0
4	174	88.94	11.06	99.65	0.35
5	168	96.55	3.45	100	0
6	167	100	0	100	0
7	148	92.94	7.06	99.79	0.21

Fig. 9 shows that the neural network correctly classifies the data in all the 7 classes of the entire dataset, with a rate of 97.2% (10588 examples out of 10889) and wrongly classifies 2.8% (301 out of 10899 examples).

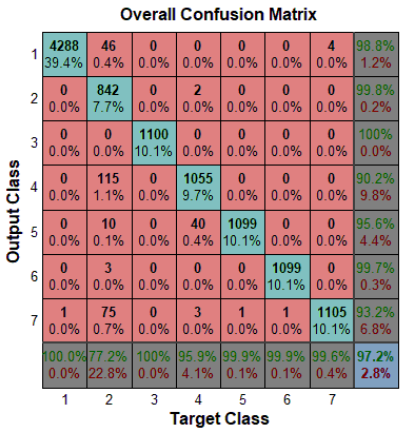

Fig. 9. The overall confusion matrix of the recognition neural network

Fig. 10 – 11 show the ROC ("Receiver Operating Characteristics") curves obtained in learning and validation, as another way to assess the quality of the pattern recognition system. The ROC result for testing can be observed in [11].

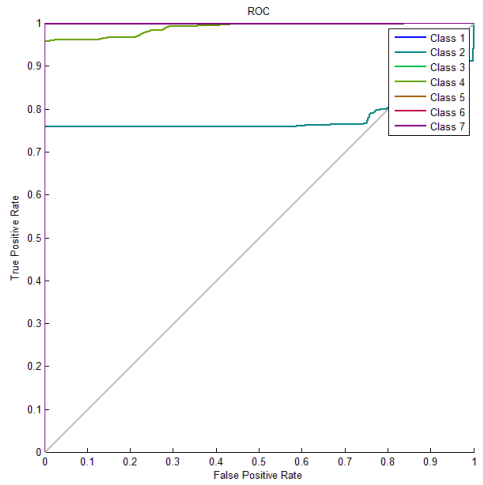

Fig. 10. The ROC curve for 7623 training data

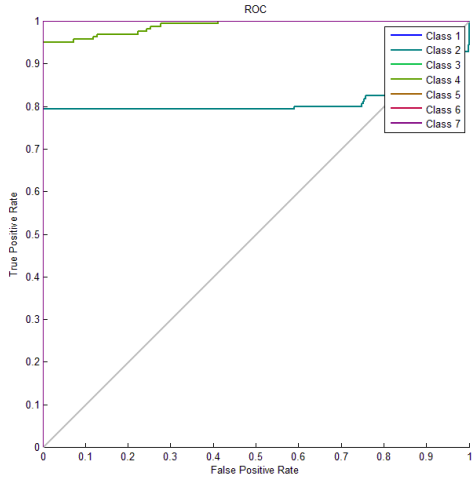

Fig. 11. The ROC curve for the 1633 validation data

The horizontal axis of the graphs indicates the FP rate (specificity), and the vertical axis represents the TP rate (sensitivity), with ranges between 0 (for 0%) and 1 (for 100%). Each class corresponds to a curve marked with a specific color that is specified in the legend. The best recognition accuracy corresponds to the curves located closer to the upper left corner. It appears that the lowest recognition rate is recorded for classes 2 and 4. The other classes exhibit a very good recognition accuracy (especially for classes 1, 3 and 6).

V. Conclusions

In this paper, a feed-forward neural network is used as a classifier, as an alternative method to recognize net or partial faults. The supervised process runs in a wastewater treatment plant and the faults can occur at the level of measurement and control equipment (sensors and actuator components). The 7 considered classes are: normal operation state, fault of recirculation pump, fault of supply pump, fault of the excess sludge pump, fault of the biomass concentration sensor, fault of the dissolved oxygen concentration sensor, partial fault of supplying pump (operating at 25% of capacity).

The architecture of the network is very simple and well known, yet it gets a good recognition performance. The overall true recognition rate of 97.2% and the rate of false classification (2.8%) prove that very good results can be obtained, using the proposed structure. The training of the neural network doesn't require a long time, but it is important that the experimental data set to be large enough and representative for all cases of considered faults.

Acknowledgment

The authors wish to thank the support received from the program "Partnerships in priority areas - PN II", implemented with the support of MECS - UEFISCDI, Project No. 269/2014 - BIOCON.

References

[1] G.M. Zeng, X.D. Li, R. Jiang, J.B. Li, and G.H. Huang, "Fault Diagnosis of WWTP Based on Improved Support Vector Machine", Environmental Engineering Science. November 2006, 23(6): 1044-1054.

[2] D. Garcia-Alvarez, M.J. Fuente, P. Vega and G. Sainz, "Fault Detection and Diagnosis using Multivariate Statistical Techniques in a Wastewater Treatment Plant", Journal: IFAC Proceedings Volumes, 2009, Volume 42, Number 11, Page 952.

[3] M.J. Fuente, D. Garcia-Alvarez, G. Sainz, P. Vega, "Fault detection in a wastewater treatment plant based on neural networks and PCA", MED 2012, Barcelona, pp. 752-757.

[4] M. Miron, L. Frangu, S. Caraman, „Actuator fault detection using extended Kalman filter for a wastewater treatment process", 21st International Conference on System Theory, Control and Computing, October 19 - 21, 2017, Sinaia, Romania, pp. 583-588.

[5] M. Miron, L. Frangu, S. Caraman, "Fault detection method for a wastewater treatment process based on a neural model," 2017 5th International Symposium on Electrical and Electronics Engineering (ISEEE), October 20 – 22, 2017, Galati, Romania, pp. 1-6, Galati Romania.

[6] F. Nejjari, V. Puig, L. Giancristofaro S. Koehler," Extended Luenberger Observer-Based Fault Detection for an Activated Sludge Process", Proc. of 17th IFAC World Congress, Seoul, Korea, July 6-11, 2008, pp. 9725 – 9730.

[7] M. Barbu, S. Caraman, G. Ifrim, "State observers for food industry wastewater treatment processes" Volume: 12, Issue: 2, Pages: 678-687, Published: 2011.

[8] J. Basu1, D. Bhattacharyya, T. Kim, "Use of Artificial Neural Network in Pattern Recognition", International Journal of Software Engineering and Its Applications, Vol. 4, No. 2, April 2010, pp. 23 – 33.

[9] K. S. Narendra and K. Parthasarathy, "Identification and Control of Dynamical Systems using Neural Networks", IEEE Transactions of Neural Networks, Vol. 1, No. 1, march 1990, pp. 6 and pp. 13.

[10] Barbu, M., Contributions to the automatic control of biotechnological processes, PhD Thesis, "Dunarea de Jos" University, Galati, Romania, Noiembrie 2006, 178 pp.

[11] Miron, M., Contributions regarding wastewater treatment plant diagnosis using neural networks, PhD Thesis, "Dunarea de Jos" University, Galati, Romania, Noiembrie 2018, 179 pp.