Impact of input layer of NAR neural network on total prediction accuracy

Vladimír Stenchlák
Department of Automation and Production Systems
University of Žilina,
Žilina, Slovakia,
vladimir.stenchlak@fstroj.uniza.sk

Ivan Kuric
Department of Automation and Production Systems
University of Žilina,
Žilina, Slovakia,
ivan.kuric@fstroj.uniza.sk

Vladimír Tlach
Department of Automation and Production Systems
University of Žilina,
Žilina, Slovakia,
vladimir.tlach@fstroj.uniza.sk

Dariusz Wiecek
University of Bielsko Biala,
Willowa 2 43-309 Bielsko Biala,
Poland,
wiecekd@ath.bielsko.pl

Keywords: Artificial Intelligence; Neural Networks; Prediction Modelling; Nonlinear Autoregressive Networks, Prediction

Abstract - This paper deals with nonlinear autoregressive neural networks. In this paper there is focus on the comparison of two neural network structures for time series prediction. The paper describes how the size of the input layer affects the overall accuracy of the prediction. A data set that is freely available online is used for time series prediction. The data set contains the measured air quality data at moderately frequented road traffic - specifically we focus on carbon monoxide concentrations given in the dataset. The NAR neural network will be in this case a model which can calculate predicted values of carbon monoxide at any time in advance.

1. Introduction

Considering how artificial intelligence algorithms like artificial neural networks, genetic or evolutionary algorithms are expanding sharply nowadays in every field of application it is important to know every single factor that can affect the performance of a given model. Neural networks are often used as data classifier or data approximator which can be useful when you want to predict some important values of required variables. The fact is that there is no standardized methodology of designing those algorithms. There are no standardized rules for designing neural networks, that is why it is important to fully understand the principles and mathematical mechanics of every type of neural networks. In this paper we only focus on NAR neural networks.

In this paper there are described nonlinear autoregressive neural networks (NAR) which are commonly used for time series (variables which are primary dependent from time) predicting. In comparison with feed-forward neural networks they have got a feedback loop – NAR networks are recurrent neural networks. The goal of this paper is to find out how the input layer of NAR neural networks affects the total accuracy of the prediction. In this paper we will compare performance of 2 NAR neural networks with different input layer size. Neural networks will be trained in MATLAB Mathworks. We will test and simulate neural networks in our standalone .net application.

2. Structure of NAR neural networks

Nonlinear Autoregressive Neural Networks – NAR NN are commonly used as time series predicting model. They can predict variables which are mainly dependent from time. In comparison with feed-forward neural networks they have feedback loop in addition. That is why they are called recurrent neural networks. All layers are the same with feed-forward neural networks except the input layer. Input layer and input neurons in NAR NN are representing time buffer of one variable [**3**]. In feed-forward neural networks each neuron represents different variable from the given time series. The input buffer layer (Figure 1). is filled with most actual values, which are calculated by the feed forward neural network.

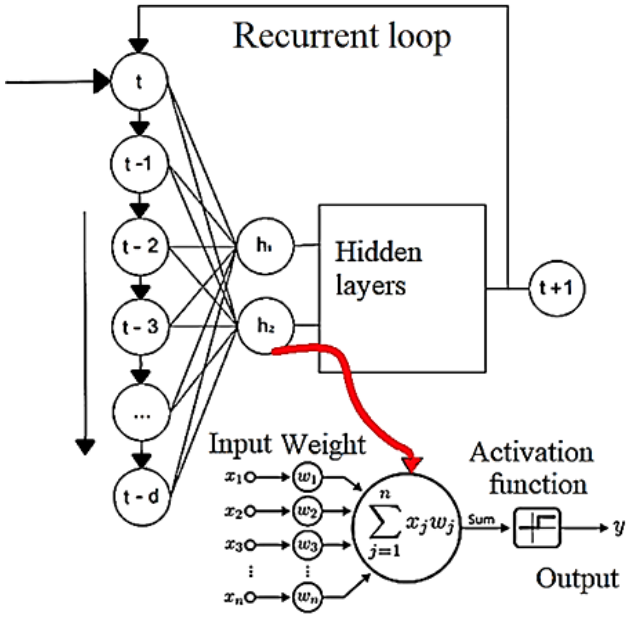

Figure 1 - Structure of NAR neural network

One artificial neuron performs 2 simple calculations [**1**]. Those calculations are the same for every type of neural network (FFNN, LSTM, CNN). Firstly, the artificial neuron needs to calculate the weighted input. This mathematical operation is given in equation (1),

$$in = \sum_{i=1}^{n} w_i x_i \quad (1)$$

where *in* is the weighted input, *n* is number of input weights connections, *w* is weight parameter value and *x* is the actual input value [**4**].

After the artificial neuron calculates the actual weighted input, this value is transferred through function which is also called activation function. There are several types of activation functions for example logical sigmoid function, hyperbolic tangent activation functions, linear function, or many others. In this article were used hyperbolic activations functions in hidden layers and linear activation function in the output layer [**2**].

Based on the input layer, the neural network itself should be possible to compute *t+1* value according to the time series input with length *d*, also called delay [**3**]. You can see the equation (2) below.

$$t + 1 = y(t, t-1, t-2 \ldots t-d) \quad (2)$$

3. Training, testing, validation data set

The biggest advantage of this type of neural network is the use of the backpropagation methods for training. Training dataset should contain expecting inputs (*t, t-1, t- 2...t-d*) and required outputs also called targets (*t+1*). In MATLAB there is possibility to choose and test many variations of backpropagation training algorithms for example Levenberg-Marquardt method, gradient descent method, Bayesian regularization method and many others. Methods shown above have different advantages and disadvantages. We used in this paper method called Scaled Conjugate Gradient [**4**].

In this paper, there were designed two neural NAR networks which will differ only in the input time delay layer. Hidden layers of both networks will be the same size. For testing the input layer size impact on the prediction accuracy, in this paper there is used data set called "Air Quality data set", which is available on Machine Learning repository for free [**5**]. This data set contains parameters as average amount of carbon monoxide, amount of carbons without methane, amount of benzene and others. Data set contains 15x9357 samples in total. The data were collected in 1 year (March 2004 – February 2005). Measuring station was deployed near the traffic road in polluted air area. We focused on one parameter which values we wanted to predict – carbon monoxide. Each sample from the data set was recorded every hour. We designed two NAR neural networks for predicting the values of the carbon monoxide. You can see on the figure 2. that the values of carbon monoxide are very complex and hard to predict by using the standard statistical methods. The two designed NAR models had different time delay input layer so comparison of the performance of those models were also made.

Corrupted samples, which haven´t got the correct information were removed from the training dataset. Corrupted data were storing value -200. Next step was to create training samples by using of programmed .net application [**8**]. The training samples were created based on the delay size of the input buffer. The first training data set was created with 10 neurons and the bigger one contained 20 neurons in input time delay buffer. The size of the training data set depends on the size of the time buffer (input layer).

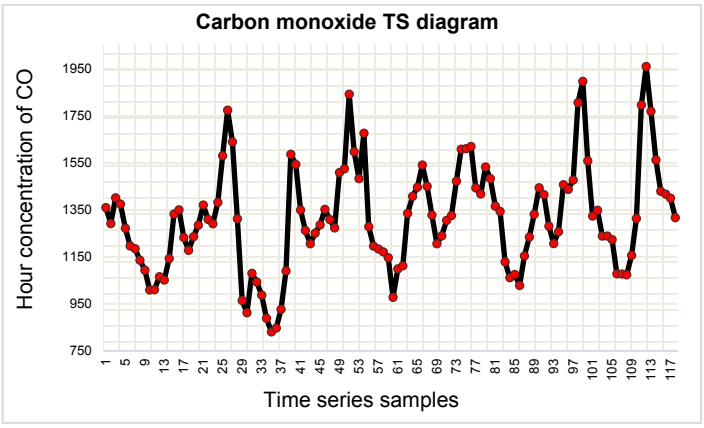

Figure 2 - Time series plot of examined carbon monoxide (117 samples)

Also, it is very necessary for optimal training of the two models divide the training dataset into training, testing and validation dataset. Training dataset is important because weight and bias parameters are adjusted according to the overall error on this dataset during the training. Validation data set is important for the information about generalization of the trained model. Testing dataset gives the independent information about the performance of our model. MATLAB uses default function for dataset division. It is called *dividerand()* and this function is dividing the dataset by the default ratio coefficients: trainRatio = 0.70, valRatio = 0.15 and testRatio = 0.15. Sum of those coefficients is equal to 1 (100% of the samples from the dataset).

4. Design and simulation of neural network

There is shown the structure of the compared neural networks in the figure below. We used *nntool* toolbox from MATLAB for creating and designing the 2 compared neural networks. We are using regular feed-forward neural networks from *nntool*. In the standalone .net application where the simulation is running, we have programmed the automatic filling of the input delay layer [**6**][**7**]. We did a computational loop, where we can calculate more prediction steps ahead values, that is how we can demonstrate the multi-step prediction.

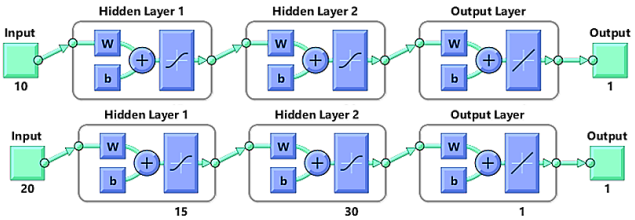

Figure 3 - Structures of designed models in MATLAB

The networks were designed as feed-forward neural networks because we are processing the input data for the NAR network in our .net application. Also, it is possible to design network with multiple hidden layers if the designing of this network is made

as feed-forward neural network in MATLAB. We used MATLAB only for setting up the bias and the weight parameters of the network to the optimal values. As you can see on the figure below after the training of the first smaller network the correlation coefficient has reached values of R = 0.857. This training took 47 iterations, and it was ended by validation check.

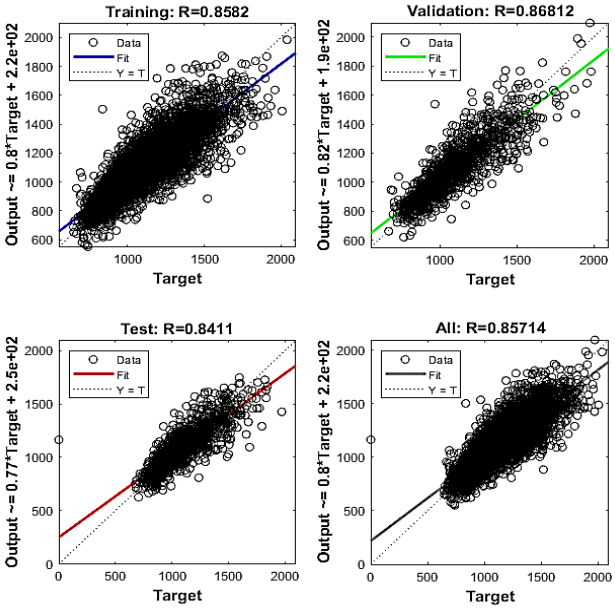

Figure 4 - Regression plot of NAR model with 10 neurons in input layer

In this paper there was made an accuracy test of the NAR NN in our standalone .net application. Our application can create a feedback loop so it can calculate multiple steps of prediction. Prediction of 1 step ahead is in this case 1 hour. The accuracy is shown in the figures below. The real data are plotted with black color. Predicted values are red points on the given plot.

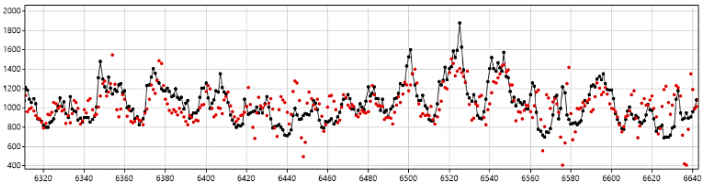

Figure 5 - 1 step ahead prediction (10 nodes in TDL)

Predicted value accuracy has significantly dropped at the 25 steps ahead prediction as you can see on the figure below. So, we can say that this model with this size of input layer wouldn´t be recommended for predicting values of more than 5 steps (hours) ahead.

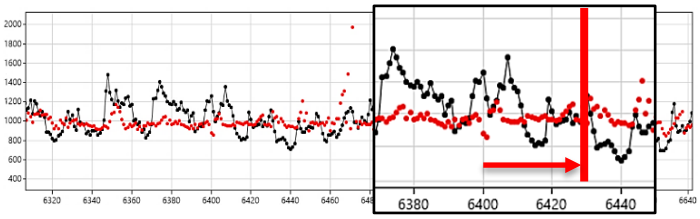

Figure 6 - 25 steps ahead prediction (10 nodes in TDL)

In Figure 6, the green rectangle area represents the input values range which comes into network input buffer layer. After filling the input layer, the network calculates 1 step ahead prediction value. This result is shifted into the input layer and the network can calculate another value which is in this case 2 steps ahead prediction value. This is how the network can calculate for example 25 steps ahead prediction values. This value is highlighted with red line in the Fig. 6.

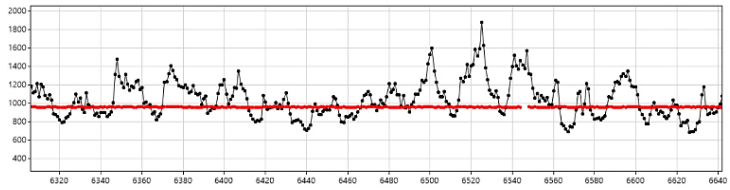

Figure 7 - 100 steps ahead prediction (10 nodes in TDL)

The other NAR NN has contained 20 input neurons in input delay layer. Two hidden layers were the same size as the first network layers were. The training took 66 iterations. In this case the training took more time than in the network with less neurons in input layer. The training was also ended by the validation check and the correlation coefficient reached value R = 0.8901. Regression plot of performance of this network is given in the figure below.

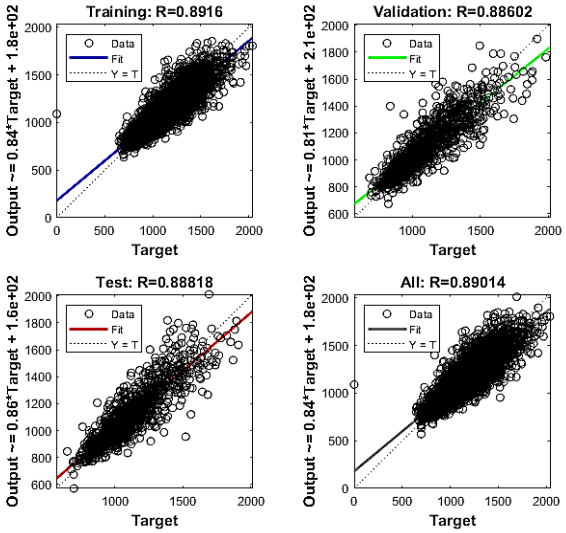

Figure 8 - Regression plot of NAR model with 20 neurons in input layer

On the figures given below this model performed slightly better than previous model with 10 neurons in the input layer. Black plotted curve represents the real data, green points are the predicted values. On the figure 9. there is a prediction performance at 1 step ahead. Because the structure of the network was bigger, the ability to approximate any function was better and the overall accuracy was higher. When neural network is approximating the function more accurately, the error of the one step prediction is not transferred and does not increase in further step predictions.

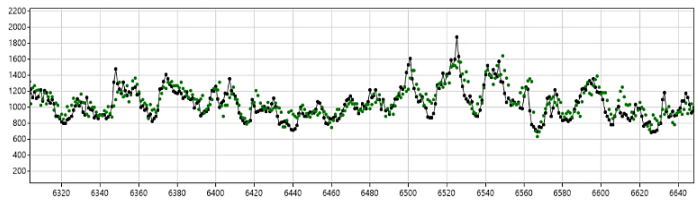

Figure 9 - 1 step ahead prediction (20 nodes in TDL)

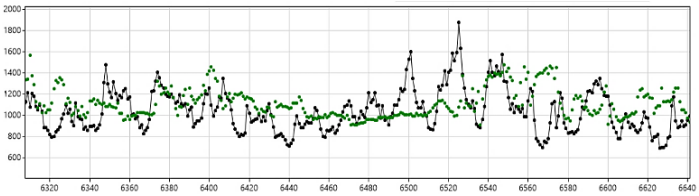

Figure 10 - 25 steps ahead prediction (20 nodes in TDL)

In the fig. 11. the input layer size is wider than in the previous network – the input values are in the green area rectangle. The red line is the 100 prediction steps ahead and it was reach by calculating and shifting the results in the input layer 100 times.

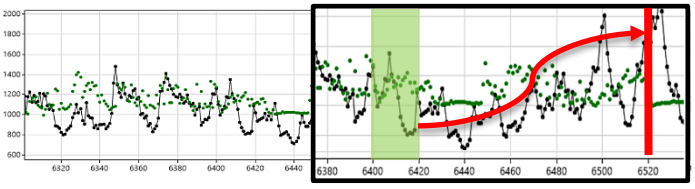

Figure 11 - 100 steps ahead prediction

5. Accuracy evaluation of multistep ahead prediction

For this accuracy evaluation is important to design bigger neural network so it can perform slightly better than in previous cases. In this case we increased the input time delay layer to 180. Also, the hidden layers were increased to 150. The idea is to calculate various statistical and mathematical metrics and based on their values the algorithm can start a new training of the NAR model when the statistical metrics reach critical defined values. There are many automation possibilities in those predictive computing algorithms.

The training of the model is ended by different pre-conditions by the one step ahead prediction calculation. In this paper there is described how parameters like correlation coefficient, coefficient of determination and root mean squared error RMSE [**9**].

RMSE is frequently used in evaluation of prediction accuracy and it is not unit-free metric. This measure is also called scale-dependent measure. It means that the accuracy is expressed in *SI* units of the examined or predicted variable. Advantage of using this statistical metric is that it gives a higher significance to higher residues and vice versa it gives a low significance to low residues. The relation between the predicted values y_i and the real values x_i in this statistical parameter is given below (eq. 3) [**9**] [**10**].

$$RMSE = \sqrt{\frac{\sum_{i=0}^{n}(x_i - y_i)^2}{n}} \tag{3}$$

Another statistic parameter which is often used for accuracy evaluation is the Mean Absolute Error MAE. It is another type of scale-dependent error, so this error uses the same units as the data being measured. Its definition is given in eq. 4. [**11**]

$$MAE = \frac{\sum_{i=0}^{n}|x_i - y_i|}{n} \tag{4}$$

Statistical parameters and metrics which units aren´t dependent on the scale is also very important. For example, there are metrics like correlation coefficient, coefficient of determination or percentual evaluated error [**12**].

In this research we used Pearson correlation coefficient r which describes the linear relationship between two variables. Random measurement error (inevitable error components) will reduce the correlation coefficient between two variables. These correlation coefficient values lay in the range of (-1, 1) [**13**]. Very important is the number of samples of the two variables n (real values x_i and the predicted values y_i) and the means of these two variables $\bar{x}, \bar{y}$. The Pearson correlation coefficient is given in eq. 5.

$$r = \frac{\sum_{i=0}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=0}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=0}^{n}(y_i - \bar{y})^2}} \tag{5}$$

When the correlation coefficient is reaching values near to 1, it means that there is a very strong linear dependency between x and y variables (line for which x increases as y increases) (more in TABLE 1.). If the correlation coefficient is reaching value -1, it means that all points lay on the line for which x increases as y decreases. Value near 0 represents that there is no linear dependency between the two variables [**14**].

Another important interpretation of this measure is coefficient of determination which in this case is r^2. This coefficient measures the proportion of the common variance. It interprets how the change of one variable affects another. When the simple linear regression is made, and the intercept is included the coefficient of determination is simply the square of the correlation coefficient r [**15**]. The range of values lays because of this fact in the interval of (0, 1).

Another goal of this article was to examine how the multistep prediction accuracy (evaluated by RMSE, r, r^2) was affected by the number of prediction steps. In this article the trained model was performed the multistep ahead prediction multiple times – 10, 30, 50, 110 and 160 steps ahead prediction. The predicted values were compared to the real values and correlation analysis was made. Also, there was made a calculation of RMSE, correlation coefficient and coefficient of determination.

The sample size on which these metrics were calculated was n = 6050 and the dependency of the RMSE error and prediction steps is given in the figure 13.

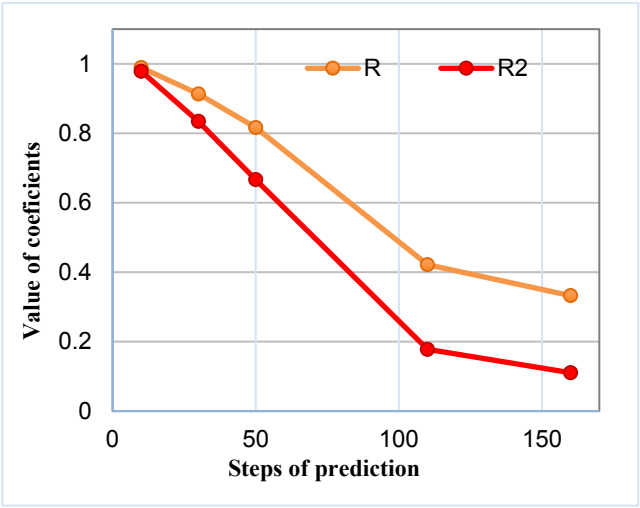

Figure 12 - Statistical coefficients dependency on prediction steps ahead (R2 – coefficient of determination, R – correlation coefficient)

TABLE 1 - Cohen's interpretation of correlation coefficient *r*

r	**interpretation of *r***
0.0 – 0.1	trivial correlation
0.1 – 0.3	small correlation
0.3 – 0.5	medium correlation
0.5 – 0.7	high correlation
0.7 – 0.9	very high
0.9 - 1	perfect

In accordance to Figure 12 and TABLE 1 [**16**], we can set up the optimal prediction steps ahead amount by the values of these coefficients. This process can be fully automated, but it depends on the application of use. Another evaluation criterion can be coefficient of determination r^2.

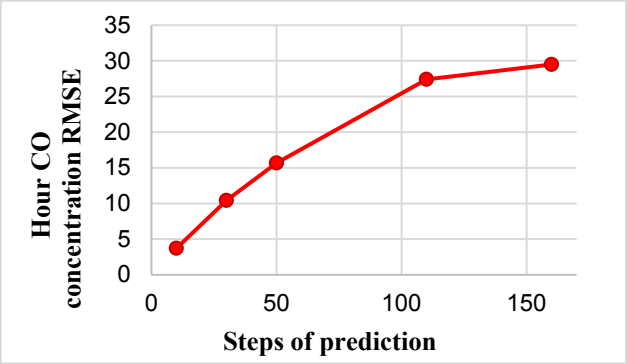

Figure 13 - RMSE and steps ahead prediction dependency

The problem or disadvantage of multistep ahead prediction is demonstrated on the regression plots below Figures 14, 15, 16. The nonlinear autoregressive models are performing well at short time prediction calculations. It is because of the error which is accumulated through the predicting iterations. There are two ways how to improve the accuracy of the model. Set up lower prediction step ahead amount or increase the structure of the model and finally improve the prediction ability (like in previous chapter). The meaning and interpretation of the correlation coefficient in accordance with Cohen is given in the TABLE 1. On the Figure 14, there is made a regression analysis between predicted values at 10 steps ahead and the real values. Coefficient of correlation reached value 0.98 and so coefficient of determination does. Error RMSE was in this case 3.71.

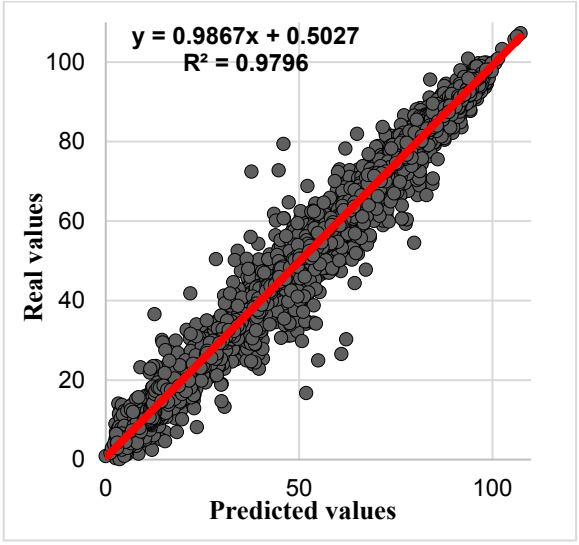

Figure 14 - Linear regression of predicted vs. real data in 10 steps ahead prediction

On the Figure 15, there is a regression analysis between predicted values at 30 steps ahead prediction and the real values at this specific time. Correlation coefficient r = 0.91, coefficient of determination reached value 0.83. RMSE = 10.43.

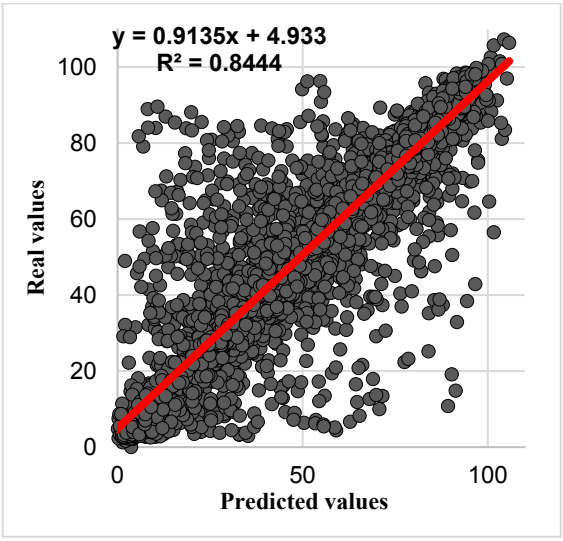

Figure 15 – Linear regression of predicted vs. real data in 30 steps ahead prediction

In the Figure 16, there is a regression analysis between predicted values at 160 steps ahead prediction and the real values at the specific time. There is a significant decrease of the correlation coefficient and the determination coefficient. Correlation coefficient r reached value of 0.33 and the coefficient of determination r^2 reached value of 0.11. Error RMSE was in this case 29.5.

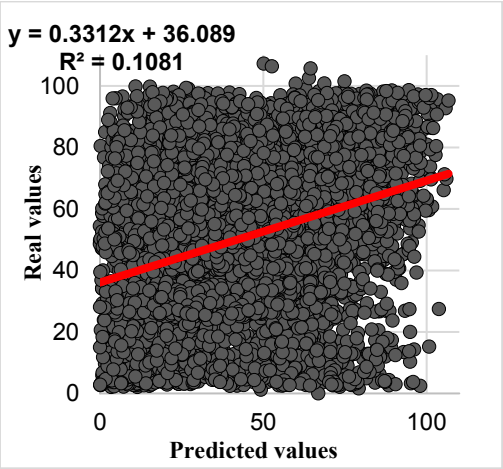

Figure 16 - Linear regression of predicted vs. real data in 160 steps ahead prediction

It is important to find an optimal setting between the step ahead prediction and the overall accuracy of the designed neural network NAR model in every application of use.

6. MATLAB and C# source codes

We created a method in C# application called *NeuralNetworkCalc()* which calculates the activation values of the output layer of a given network. The input argument for this method is an input layer array with the input values for the network. For every *step+1* ahead prediction it is necessary to buffer the input layer of the network with previous result of the prediction. This method in C# was written by use of three nested for loops cycles as shown below (for one step ahead prediction):

```
for(int i=1;i<neurons.Length;i++)
{
 for(int j=0;j<activations[i].Length; j++)
 {
   for(int k=0;k<activations[i-1].Length;k++)
   {
activations[i][j]=activations[i][j]+activations[i-
1][k]*weightLayers[i-1][k,j];
   }
   switch(activationFunctions[i-1])
   {
     case 0:
activations[i][j]=Math.Tanh(activations[i][j]+biases
[i-1][j]);
     break;
     case 1:
activations[i][j]=activations[i][j]+biases[i-1][j];
     break;
   }
 }
}
double[] output = new double[output.Length];
output = activations[activations.Length-1];
return output;
```

The weight layers values in the previous code block are loaded and exported from Matlab after the training. The network needs to have a defined minimum and maximum values from the training dataset. This was necessary because of the data normalization purposes which MATLAB does automatically [**17**]. The normalization function which MATLAB uses called *minmax* is given in the C# source code bellow:

```
double result;
result=(((maxY-minY)*(input-minX))/(maxX-
minX)+minY;
return result;
```

If we want to interpret the results from the network in the units in which the actual quantity is given, we need to use inverse function of this normalization function *minmax()*.

If there is a need to avoid using the MATLAB Neural Network toolbox GUI there is a way using MATLAB script language to create, train and simulate trained neural network on custom input specific values. There is a given example how to create a neural network with specific layer size and activation functions.

```
simulationNet=network;
simulationNet.numInputs=1;
simulationNet.inputs{1}.size=8;
simulationNet.numLayers=4;
simulationNet.outputConnect=[0 0 0 1];
simulationNet.layers{1}.size=150;
simulationNet.inputConnect(1,1)=1;
simulationNet.biasConnect=[1;1;1;1];
simulationNet.layers{1}.transferFcn='tansig';
view(simulationNet);
```

For the data preparation it was necessary to create an input-target training data set from the time series data. This training data set is dependent on the neural network input and output layer size. Also, it is necessary to have the data set for processing in MATLAB in ascending time order.

7. Conclusion

Considering the performance of each neural network which was trained, it is evident that enlarging the input layer size increases the accuracy of multistep ahead value prediction. The reason why the bigger neural network was performing better and more accurately in multistep ahead prediction was that neural networks with larger architecture and bigger size of each layer can approximate required function more accurately. Disadvantage of bigger neural networks is the longer training. If the model has larger input layer, it can search for historical anomalies in input data and so it can better approximate the given function. It is very important to have low-error response in one step ahead prediction. If this error is high, the total accuracy of the model in multistep ahead predictions are distorted by this error.

After training it is necessary to check the multistep ahead prediction and find the optimal steps ahead parameter, so the error response of this model is sufficient. In this paper in chapter V. there was trained a bigger NAR model and the one step ahead prediction was very accurate in this case. There is dependency analysis between correlation coefficient r and coefficient of determination r^2. Also, there was a comparison between RMSE error and the prediction step ahead calculation.

Also, it is important to consider that these results from this paper may differ depending on the data processed and used in the actual prediction task. Also, this paper is aimed only on Nonlinear Autoregressive Neural Networks. Some other architectures for

example LSTM or NARX neural networks can perform better in prediction tasks than NAR neural networks. The prediction accuracy is also dependent on the signal which is predicted. Complex waveforms of predicted variables are more difficult to predict in comparison with variables which are not so often changed over the time.

Acknowledgment

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-16-0283. Project title: Research and development of multi-criteria diagnosis of production machinery and equipment based on the implementation of artificial intelligence methods.

References

[1] SESTILLI, Carson.: Deep Learning: Going Deeper toward Meaningful Patterns in Complex Data: https://insights.sei.cmu.edu/sei_blog/2018/02/deep-learning-going-deeper-toward-meaningful-patterns-in-complex-data.html, 10.3.2020

[2] SHARMA, Avinash. Understanding Activation Functions in Neural Networks: https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0, 1.3.2020

[3] POTOCNIK, Primoz. Prediction of chaotic time series with NAR neural network: http://lab.fs.uni-lj.si/lasin/wp/IMIT_files/neural/nn05_narnet/, 2.4.2020

[4] NIELSEN M.: A visual proof that neural nets can compute any function: http://neuralnetworksanddeeplearning.com/chap4.html

[5] S. DE VITO, E. MASSERA, M. PIGA, L. MARTINOTTO, G. DI FRANCIA, On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario, Sensors and Actuators B: Chemical, Volume 129, Issue 2, 22 February 2008, Pages 750-757, ISSN 0925-4005

[6] KOSINAR, M.; KURIC, I. Monitoring possibilites of CNC machine tools accuracy. In: Proceedings of 1st International Conference on Quality and Innovation in Engineering and Management (QIEM), 17-19.3.2011, p. 115-118

[7] SAPIETOVA, A., M. SÁGA, I. KURIC et al., Application of optimization algorithms for robot systems designing. International Journal of Advanced Robotic Systems, 2018, ISSN: 1729-8814

[8] KURIC, I.; CISAR, M.; TLACH, V.; et al..: Technical Diagnostics at the Department of Automation and Production Systems. Book Series: Advances in Intelligent Systems and Computing, Volume: 835, p. 474-484, Published 2019

[9] ARMSTRONG, J. SCOTT; COLLOPY, Fred (1992). "Error Measures For Generalizing About Forecasting Methods: Empirical Comparisons" (PDF). International Journal of Forecasting. 8 (1): 69–80. CiteSeerX 10.1.1.423.508.

[10] LEHMANN, E. L.; CASELLA, George (1998). Theory of Point Estimation (2nd ed.). New York: Springer. ISBN 978-0-387-98502-2. MR 1639875.

[11] WILLMOTT, Cort J.; MATSUURA, Kenji (December 19, 2005). "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance". Climate Research. 30: 79–82. doi:10.3354/cr030079

[12] TAYLOR, John R. (1997). An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements (2nd ed.). Sausalito, CA: University Science Books. p. 217. ISBN 0-935702-75-X.

[13] FRANCIS, DP; COATS AJ; GIBSON D (1999). "How high can a correlation coefficient be?". Int J Cardiol. 69 (2): 185–199. doi:10.1016/S0167-5273(99)00028-5.

[14] FROST, Jim. (2018). „Interpretation of Correlation Coefficient“. Internet article. < https://statisticsbyjim.com/basics/correlations/>

[15] Devore, Jay L. (2011). Probability and Statistics for Engineering and the Sciences (8th ed.). Boston, MA: Cengage Learning. pp. 508–510. ISBN 978-0-538-73352-6.

[16] COHEN, Jacob; et. al. (2013). „Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences“ (3rd edition). Routledge, New York. ISBN: 9780203774441

[17] DEMUTH, H., BEALE M., Neural Network Toolbox: For use with MATLAB, version 4. (SW documentation). 2004. <online> http://128.174.199.77/matlab_pdf/nnet.pdf