DOI 10.33111/nfmte.2021.067

TIME SERIES FORECASTING OF AGRICULTURAL PRODUCT PRICES USING ELMAN AND JORDAN RECURRENT NEURAL NETWORKS[1]

Tetiana Kmytiuk

Vidzeme University of Applied Sciences
4 Cesu Street, Valmiera, LV—4200, Latvia
ORCID: 0000-0001-5262-856X, E-mail: tatyanakmutyk@gmail.com

Ginta Majore

Vidzeme University of Applied Sciences.
4 Cesu Street, Valmiera, LV—4200, Latvia
ORCID: 0000-0002-9514-7229, E-mail: ginta.majore@va.lv

Most practical problems of forecasting time series are characterized by a high level of nonlinearity and nonstationarity, noise, the presence of irregular trends, jumps, and anomalous emissions. Under these conditions, statistical and mathematical assumptions limit the possibility of applying classical forecasting methods. The main disadvantage of statistical models is the difficulty of choosing the type of model and selecting its parameters. An alternative to these methods may be methods of computational intelligence, which include artificial neural networks, which can significantly improve the accuracy of time series prediction. A significant advantage of neural networks is that they are able to learn and generalize the accumulated knowledge, highlighting the hidden relationships between input and output data. At the moment, the most time series forecasting solutions based on this toolkit involve the use of feed-forward neural networks (perceptrons, convolutional neural networks, etc.). The article provides an overview of the architecture, principles of operation, and methods of teaching known models of recurrent neural networks. In the study, we built and compared the architectures of Elman and Jordan neural networks for solving the problem of forecasting prices for agricultural products. The corresponding statistical comparisons of the above models are also given. The experimental results show that such approach provides high accuracy in predicting the values from the price of agriculture products.

Keywords: *forecasting, agricultural product prices, recurrent neural network, Elman network, Jordan network*

JEL classification: C18, C45, C53, Q11

[1] The article came into being within research grant from State Education Development Agency of the Republic of Latvia conducted by Tetiana Kmytiuk in Sociotechnical Systems Engineering Institute of Vidzeme University of Applied Sciences in the academic year 2021/2022

Introduction

The agricultural market is very volatile and unpredictable in nature. There are many factors that affect the prices of agricultural products, such as supply and demand, market trends, seasonality, historical prices, weather conditions, etc., all of which can lead to an increase or decrease in the number of buyers and so on. It is difficult to analyze such many factors to determine the best indicators that influence the price of products and allow predicting it. So, forecasting prices for agricultural products is a very complex process. Therefore, researchers are constantly looking for a reliable method for time series forecasting.

Time series forecasting is an important data analysis technique aimed at studying historical data, which makes it possible to predict future values based on the construction of an appropriate model that describes the internal structure of the series. Traditional methods of time series analysis, described in particular in [1], have gained wide popularity among many researchers.

Theoretical reviewing of the research advances and trend of agricultural product price forecasting methods in recent years is provided in [2]. In the article [3] the main models that are used to analyze the agricultural sector, medium and long-term forecasts, as well as agripolicy formation are described. Dabin et al. [4] proposed a model selection method based on time series features and forecast horizons for forecasting agricultural commodity prices. For forecasting in agriculture, advanced artificial intelligence tools were also used. Thus, based on the theory of fuzzy logic, an economic and mathematical model of forecasting the level of economic and administrative stimulation of agricultural production has been developed in [5].

Nevertheless, the tools mentioned above have certain shortcomings: high mathematical complexity and a huge dependence on specific knowledge about the choice of a particular model. That is why, in recent years, the artificial neural networks (ANNs) have been applied in many areas an alternative to other modelling methods. References [6-8] reveal different time series forecasting by ANNs methods.

ANN can be considered as a directed graph with weighted connections in which artificial neurons are nodes. ANNs can learn from models and discover hidden functional connections in certain data, even if they are unknown or difficult to identify. Artificial neural network has proven to be an effective model of prediction due to its

inherent properties, namely: adaptive learning, self-organization, generalization, real-time computing, and resilience. It solves the problems of traditional methods by making better use of the nonlinear, nonstationary and oscillatory nature of time series.

Generally, the neural networks that are applied for time series forecasting are divided into two types of architecture, namely feed-forward and recurrent neural network. Herewith, the vast majority of applications of neural nets for time series forecasting are based on feed-forward networks (perceptrons, convolutional neural networks, etc.).

Recurrent neural network (RNN) is a type of advanced artificial neural network, with the main difference in adding a connected neural layer between input and output vectors, which provides a hidden or output network pattern to provide feedback as input to create the next network output [9].

The main feature of recurrent neural networks is the property of memory as opposed to feed-forward networks. Due to their structure, recurrent neural networks are organically suitable for modelling sequences, including time series [10].

The advantage of the RNN method is the lower values of the errors of the forecasting than the feed-forward, although the learning time is slightly longer.

In this paper, the special recurrent neural network as Elman neural network (ENN) and Jordan neural network (JNN) are used as machine learning technology to analyze and predict future prices for agricultural products based on historical prices.

The aim of the article is to experimentally test the level of accuracy of forecasting the price of agricultural products (on the example of potatoes) using two types of RNN, such as Elman and Jordan, required for use by agricultural enterprises in practice.

Methodology

The structure and principle of operation of Recurrent Neural Networks

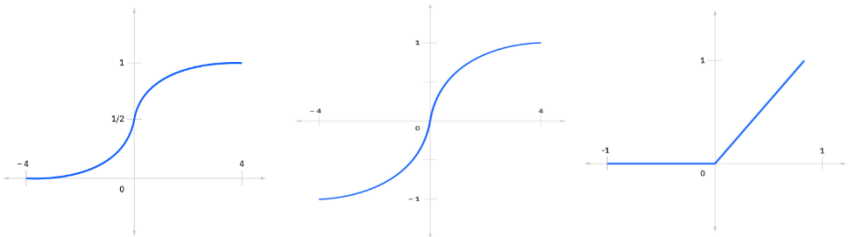
Recurrent neural networks (RNN) are a class of artificial neural networks in which connections between nodes form a time-oriented graph. This creates an internal state of the network, which allows it to exhibit dynamic behaviour over time. Unlike artificial neural

networks of feed-forward type, RNN can use their internal memory to process arbitrary sequences of inputs [11].

There are different types of architectures of RNN: one to one (have only one input and one output), one to many (single input and multiple outputs), many to one (multiple inputs and single output) and many to many (multiple inputs and multiple outputs). A key feature of RNN is that the network has feedback (reserve unit or context layer), unlike a traditional feed-forward neural network. This allows the RNN to simulate the effect of early parts of the sequence on the later part, which is a very important feature when it comes to modelling time series sequences. Another distinctive feature of recurrent networks is that they have common parameters at each network level. While feed-forward neural networks have different weights for each node, recurrent neural networks have the same weight parameter in each network layer. However, these weights are still adjusted in the process of back propagation to facilitate reinforcement training [12].

An important part of neural network design is choosing an activation function that performs a non-linear transformation on the input before sending it to the next layer of neurons or to the output of the network. The role of the activation function in the hidden layer is to adapt the neural network to the training data set. The choice of activation function at the output level will determine the type of predictions the model can make. The activation function is typically chosen based on the type of neural network architecture and can defined from three common types as follows (Fig. 1) [13]:

- sigmoid function: $sig(x) = \frac{1}{1+e^{-x}}$;
- hyperbolic tangent function: $tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$;
- rectified linear function: $relu(x) = \max(0, x)$.



sigmoid hyperbolic tangent rectified linear

Fig. 1. Different types of the activation functions

The well-known recurrent networks are Elman and Jordan neural networks. They are very similar. The only difference is that the context (delay) neurons in Elman neural network are fed from the hidden layer, instead as Jordan neural network – from output layer. Both neural networks are useful for predicting time series observations, which have a short memory.

Before an RNN can be used to solve any problem, it must be learned on the appropriate data. Learning is the process of adjusting network parameters, which play a key role within the prediction phase in RNN. After computing the output by the neural network, a corresponding desired value is transmitted to the output and an error is calculated as the difference between the target and the system output. The resulting error represents an estimation of the predictions quality of the trained network and is sent back to the system to adjust its parameters. Thereby, this process is repeated in such a way that the obtained output and the desired one are as close as possible. This formalism is called the training (learning) process.

The training aim is to modify the RNN parameters to minimize the error between the network's prediction and the actual values. There are several steps of the training algorithm of the neural network, which are as follows:

Step 1. The normalization of the input data in the RNN model. There are various methods of normalization, the main idea of which is to include normalized data in the range [0, 1]. The following equation was used in this study:

$$x'(t) = \frac{x(t) - \min(\mathrm{X})}{\max(\mathrm{X}) - \min(\mathrm{X})}, \quad (1)$$

where $\mathrm{X} = \{x(t)\}$.

Step 2. Submission of the first-time value (signal) to the network input layer.

Step 3. Computation of the states of hidden layers' neurons using the set of current inputs and previous states of neurons.

Elman Neural Network (ENN)

Elman neural network is a type of neural networks obtained from a multilayer perceptron with the addition of a context layer. The

simplest recurrent neural network that consists of one hidden and one context layer of neurons was introduced by Elman in 1990 [14].

The Fig. 2 illustrates the architecture of Elman neural network, which consists of four layers: the input layer, the hidden layer, the context layer and the output layer. The number of neurons in the inputs layer of ENN is equal m and in the hidden layer is n, and has one output unit. The hidden layer is interconnected fully with the context layer, so the number of neurons in context and hidden layers is the same. The Elman NN architecture allows you to take into account the background history of the observed processes and accumulate information to choose the right control strategy.

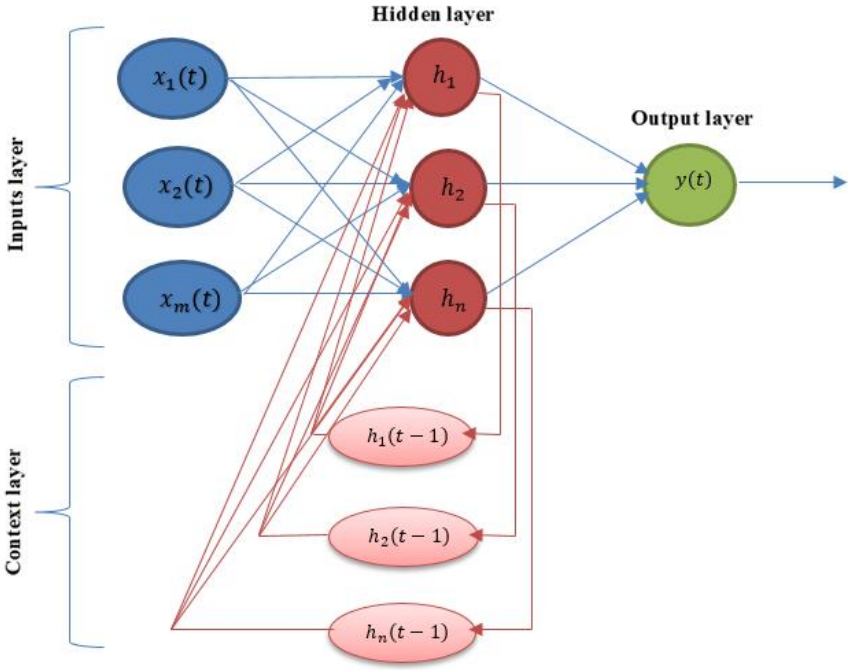


Fig. 2. Architecture of Elman recurrent neural network

A feed forward network to each other connects the input, hidden, and output layers of the Elman model. Feedback connection appears between the context and hidden layers of neurons, which is an extension of the input layer [15]. At each moment of time (each time

the data is passed to the neurons of the input layer), the neurons of the context layer maintain the previous values of hidden layer's neurons and pass them to the respective neurons of the hidden layer (to context layer). For example, the first signal (vector of inputs $\{x_i(t), i = \overline{1,m}\}$) enters the input layer and activates the neurons of the hidden layer (2). After the transformation by the hidden layer, the signal will go to the output, and its copy will be delayed in the context layer, memorizing the information. From the outputs of the neurons of the hidden layer, the signal is transmitted to the inputs of the neurons of the output layer, which form the output signal of the network (3). Then it enters the network the next signal, and a copy of the previous state of hidden layer's neurons (which memorized in neurons of context layer) is received at the same time. The mathematical of the ENN is described in [16, 17] for hidden layer as

$$h_j(t) = F\left(\sum_{i=1}^{m} w_{ij}\, x_i(t) + \sum_{l=1}^{n} w_{lj}^{`}\, h_l(t-1)\right), \qquad j = \overline{1,n}, \tag{2}$$

and for output layer as

$$y(t) = F\left(\sum_{j=1}^{n} w_j\, h_j(t)\right), \tag{3}$$

where w_{ij} is the weight between the input x_i and the j^{th} hidden unit; $w_{lj}^{`}$ is the connection's weight between the l^{th} neuron of context layer (past value at time t-1 of l^{th} neuron of hidden layer) and the j^{th} hidden unit; F is the sigmoid function; w_j is the weight of connect between the node h_j of the hidden layer with the output.

Jordan Neural Networks (JNN)

Jordan neural network is a kind of recurrent neural networks that is obtained from a multilayer perceptron if, in addition to the input vector, an output one is fed to its input with a delay of one or several clock cycles [18]. This model is very similar to Elman RNN except that the context layer stores a copy of the output layer's previous

values instead the hidden one. The Fig. 3 illustrates the architecture of the Jordan recurrent neural network [19].

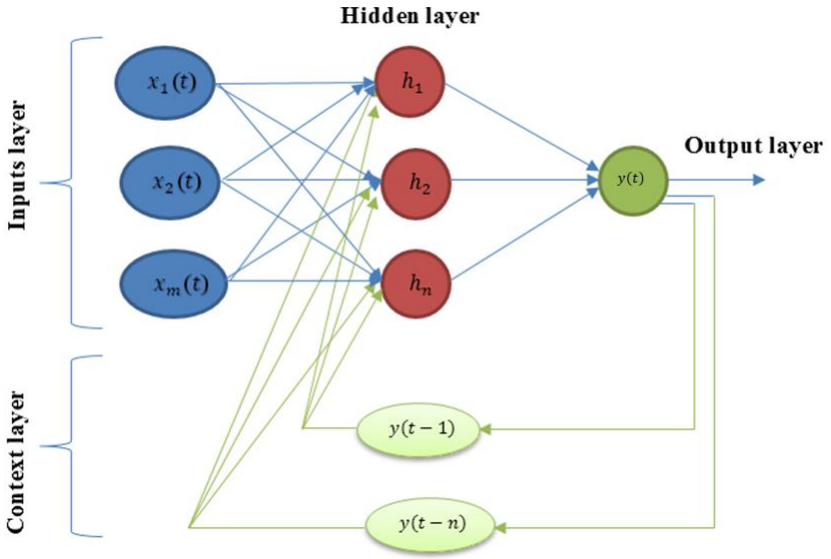


Fig. 3. Architecture of Jordan recurrent neural network

The sequence of signals passing through the Jordan neural network has the following form: the first signal enters the input layer, then goes to the hidden layer; the signal transformed by the hidden layer (4) will go to the output layer, and the resulting calculation of network (5) will be delayed in the context layer; the next signal enters the network and, together with the output image from the previous delay step on the context layer, is sent to the hidden layer, where the calculation is performed and transmitted to the output layer. The behaviour of the Jordan recurrent network can be described as a dynamic system using a pair of nonlinear matrix equations that differ from (2) and (3) by the context level [20]. So, calculations in JNN on the hidden layer are carried out according to formula

$$h_j(t) = F\left(\sum_{i=1}^{m} w_{ij}\, x_i(t) + \sum_{l=1}^{n} \grave{w}_{lj}\, y(t-l)\right), \quad j = \overline{1,n}, \tag{4}$$

and on the output layer according to

$$y(t) = F\left(\sum_{j=1}^{n} w_j\, h_j(t)\right), \tag{5}$$

where $w_{lj}^{`}$ is the connection's weight between the l^{th} neuron of context layer (network's output past value at time *t-l*) and the i^{th} hidden unit.

The structure of the Jordan neural network has the main disadvantage compared to the similar Elman neural network that the number of feedbacks is determined by the depth of the memorized time series, which is difficult to estimate in advance.

Training the RNN

The Elman and Jordan neural networks building process for prediction of time series data will be included the following steps:

Step 1. Data collecting and exploring.

This is the initial stage of the research at which the goal is determined, data collection is carried out, their type and other features are recognized, which allow a better understanding of their nature. In addition, the relationships between different data variables, the structure of the data set, the presence of outliers, and the distribution of data values are decided. Moreover, the justification of the mathematical form of calculations is carried out and the type of model is determined, in our case it is recurrent neural networks.

Step 2. Training the model on data.

Neural network learning is an iterative procedure of processing reference data sets in order to minimize the error criterion. It is common practice to split the available data into two parts – training and test data, where the training data is used to estimate any parameters of the model and the test data is used to evaluate its accuracy. The training sampling is randomly selected from the total number of observations, and the rest is left for the test sampling. At this step, a decision is made about the number of units in the hidden layer and the availability of feedback, and the model have been optimizing.

Step 3. Evaluating model performance.

The performance of a model is quantified using established model evaluation methods, which allow one to evaluate the performance of a

single model or to compare different models. Studying the results of the model is also possible through the construction of an error plot, which visualizes the relationship between the actual target values and the calculated values.

Step 4. Conclusion summed up and decisions are made to refine or use the model in practice.

Model prediction accuracy

A common way to determine the accuracy of the forecasting model is to use statistical measurements. There are numerous criteria by which we can evaluate and compare models. We decided to determine the quality of the forecast obtained by Elman and Jordan neural networks by computing the mean square error (*MSE*), root mean squared error (*RMSE*) and mean absolute error (*MAE*) [21-23]. The corresponding formulas for calculating the criteria are given in the Table 1.

Table 1

STATISTICAL EVALUATION CRITERIA

Criteria	Index	Formula
Mean Square Error	*MSE*	$\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2$
Root Mean Squared Error	*RMSE*	$\sqrt{MSE}$
Mean Absolute Error	*MAE*	$\frac{1}{n}\sum_{t=1}^{n}\lvert y_t - \hat{y}_t\rvert$

It should be noted criteria measure the deviation between the actual and predicted values, so the prediction performance is better when the values of these evaluation criteria are smaller.

Results and discussion

The prediction process starts from the preparation of the data, which was used as data on average retail prices of the potatoes in Latvia. The data were taken over a range of 16 years and 7 months, from January

2005 to July 2021 [24]. Our data set consists of monthly potatoes price so there were 199 data records used for the prediction process.

All data analyses were conducted by using R software. The decomposition of the time series was performed with the function stl in package stats. Elman and Jordan recurrent neural networks were built with the functions $elman$ and $jordan$ in the $RSNNS$ package, respectively [25].

Fig. 4 shows the visualization of time series plot of the data. We can observe a general upward trend and strong seasonality in the original time series.

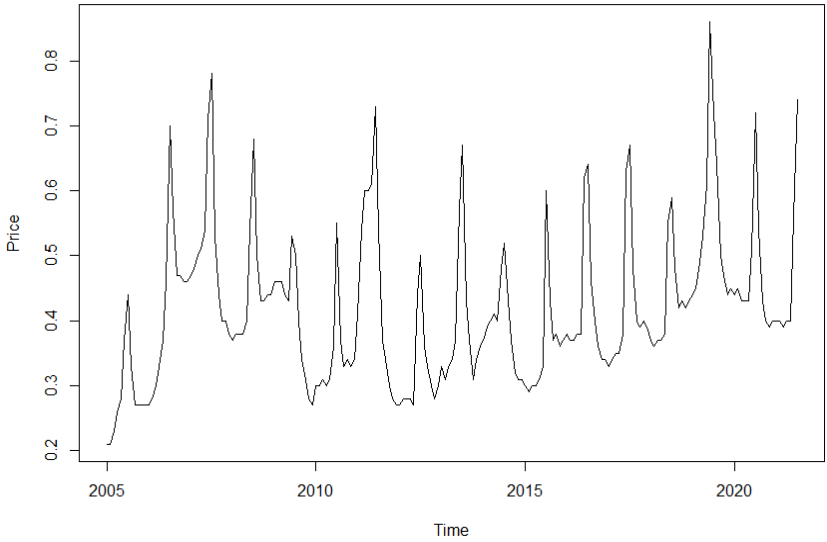


Fig. 4. Time series data of the average retail prices of the potatoes (euro per 1 kg) from Jan. 2005 to Jul. 2021

We decomposed the time series down into the main systematic components for a more detailed analysis: trend, seasonality, and random fluctuation features (see Fig. 5).

Analyzing the graph in Fig. 5, we can conclude that the period of seasonal fluctuations is 12 months. The highest price is observed in the summer months. Given the monthly frequency of the data, we created 12 time-lag variables as input features for building neural network.

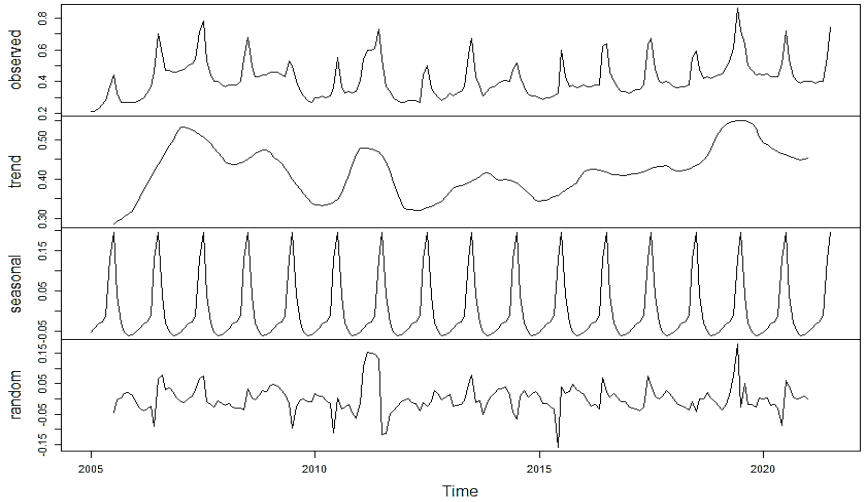


Fig. 5. Visual summary of the decomposition of time series data

The collected dataset was divided into two samplings – training and test. We adopted a common practice where 70% of the dataset was used for training and the remaining 30% was used for testing.

We built the recurrent neural network with three layers: input, hidden and output. We have done the experiment repeatedly on the training sampling on different index data and different numbers of neural nodes in the hidden layer are chosen as the optimal. We chose the optimum number of neurons by trial and error: 12 neurons (i.e. we fed into the network the past 12 values of the time series) in the input layer and 2 neurons in the hidden layer. The output layer has one neuron that get the value of the time series at time t+1. In addition, the built Elman recurrent neural network has 2 neurons in the context layer, while Jordan recurrent neural network – only one. Table 2 shows optimal parameters of the Elman and Jordan neural networks as a result of the experimental study.

The learning rate was chosen equally to the two learning process models since the ENN and JNN models have similar topology structures. ENN and JNN were trained for 1000 iterations with the learning rate of 0.1. The error function graphs of both models are illustrated in Figs. 6 and 7.

Table 2

TRAINING DATASET PARAMETERS FOR ELMAN AND JORDAN NEURAL NETWORKS

Parameter	Value
Number of neurons of input layer	12
Number of neurons of hidden layer	2
Number of neurons of output layer	1
Training sampling	70% of the data
Test sampling	30% of the data
Number of iterations	1000
Learning rate	0.1

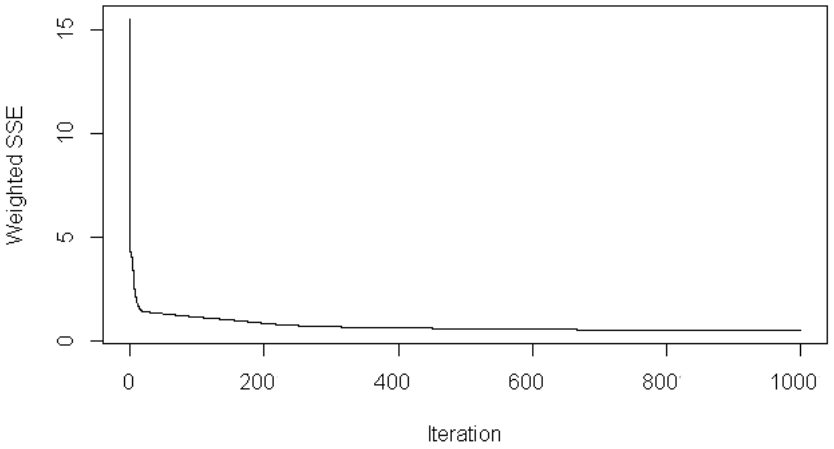


Fig. 6. Elman neural network error curve

Fig. 6 shows the decrease of Elman network error during training iterations. Given the relatively small size of the dataset, the error of the model falls to a minimum fairly quickly. Moreover, the error drops sharply within the beginning iterations. It then declines at a more modest rate until the 1000 iteration.

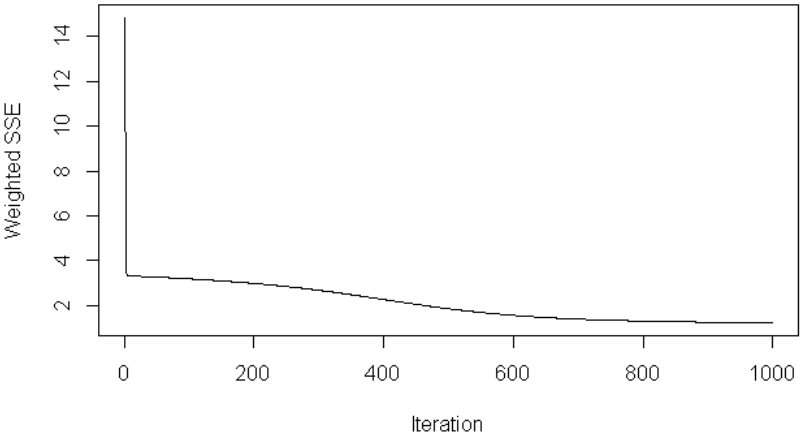


Fig. 7. Jordan neural network error curve

Regarding JNN, the error falls sharply within the first 100 and it is stable by around 500 iterations (see Fig. 7).

Plots above show that the error decreases with the epochs increase, which is a good sign of the effectiveness of the training process.

In the next step, we estimated the error of both neural network models using the *plotRegressionError* function. The visualization of the relationship between the actual target values and the predicted ones by the Elman neural network is shown in Fig. 8.

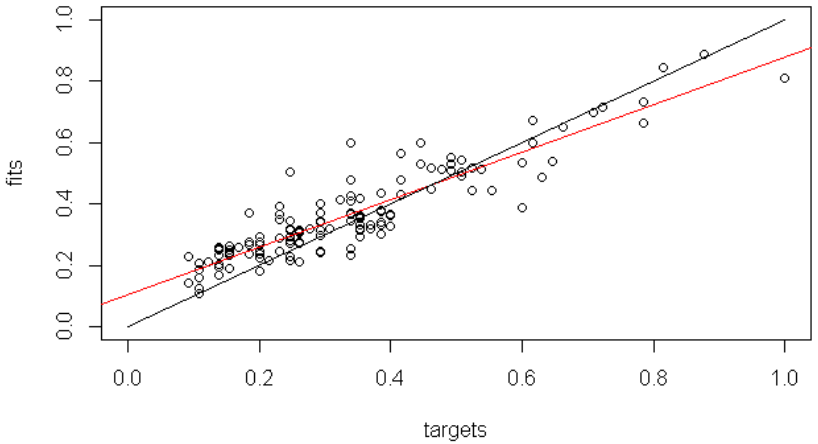


Fig. 8. Calculated by ENN and actual values

In Fig. 8 the target values (actual observations) lie on the x-axis and the calculated values for the training set lie on the y-axis. Since the red linear fit is close to the optimal (black line), this indicates an overall good fit of the training data.

We used a quantitative estimate to confirm the above, such as calculating the square of the correlation coefficient that equal $R^2 = 0.83$, that is relatively high and demonstrate strong relation between predicted and actual values.

Fig. 9 shows a linear fit to the actual data and the calculated values of Jordan neural network. We see no optimal fit since each point is not close to the estimated line, this tells us that the model doesn't good fit data. Nevertheless, we continue to use and estimate this type of recurrent neural network because the square of the correlation coefficient is more than 50% and equal $R^2 = 0.59$.

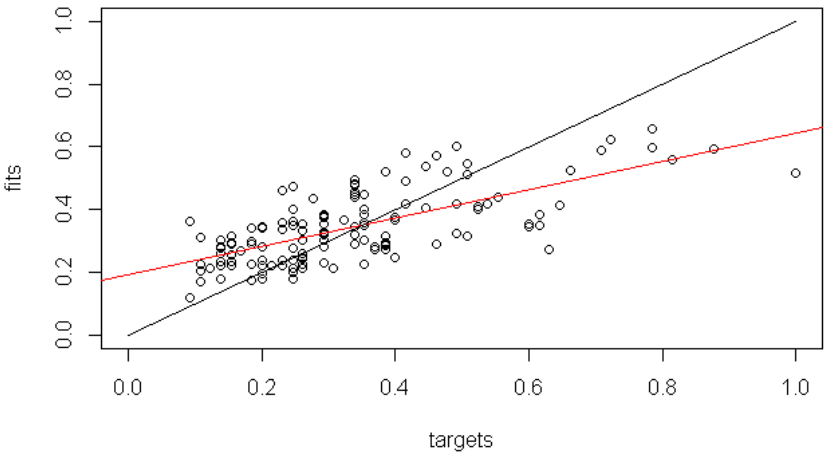


Fig. 9. Calculated and actual values of JNN

After building the model, we must verify it on test data. Scatter plots of the prediction and actual values are shown in Fig. 10 for Elman and Jordan neural networks.

We can see in Fig. 10 that the approximation is good for Elman neural network when we superimpose the prediction over the original series. This is confirmed by the square of the correlation coefficient.

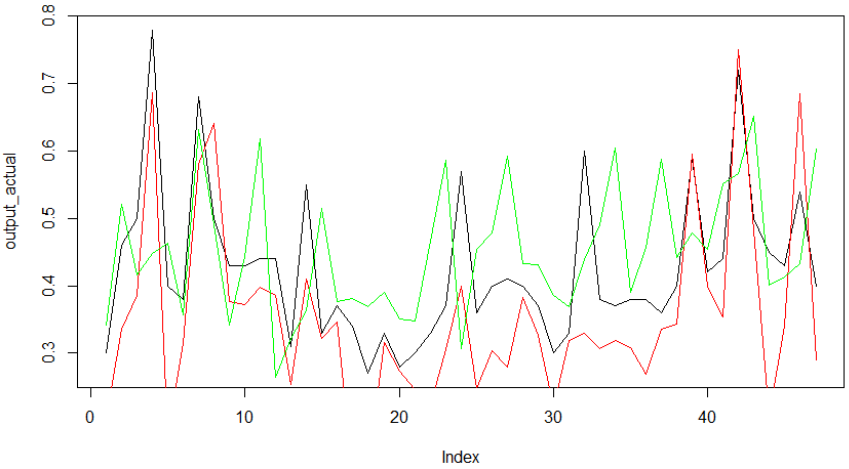


Fig. 10. Actual (black line), predicted by ENN (green line) and by JNN (red line) values from test sampling

The correlation for test sampling is quite respectable at around 0.78. Key measurements such as *MAE*, *MSE*, *RMSE* are used to evaluate the Elman and Jordan neural networks. The values of these indexes are in the tables below. Table 5 shows estimation results for training sampling and Table 6 for test one.

Table 5

COMPARISON OF PERFORMANCE OF ELMAN AND JORDAN NEURAL NETWORKS FOR TRAINING SAMPLING

Model	***MAE***	***MSE***	***RMSE***
Elman RNN	0.1522668	0.04062864	0.2015655
Jordan RNN	0.1589558	0.04136592	0.2033861

Table 6

COMPARISON OF PERFORMANCE OF ELMAN AND JORDAN NEURAL NETWORKS FOR TEST SAMPLING

Model	***MAE***	***MSE***	***RMSE***
Elman RNN	0.204134	0.060927	0.246834
Jordan RNN	0.215763	0.068208	0.261166

Tables 5 and 6 show that the evaluation criteria of the Elman RNN are smaller than of the Jordan model both for training and test samplings. From Table 6 and Fig. 10 we can conclude that the proposed ENN model is effective and accurate for time series forecasting. However, both Elman and Jordan recurrent neural networks can be used for forecasting nonlinear time series data, for example in agriculture sphere.

Conclusion

This article is devoted to solving the problem of predicting historical prices for agricultural products using the example of potatoes and studying the applicability of Elman and Jordan recurrent neural networks in modelling such time series. We found that even using a randomly selected sample, these neural networks deliver a strong performance in terms of the *MAE*, *MSE*, and *RMSE* statistical criteria for both samplings: training and test.

It should be noted that when developing a neural network to solve a particular problem, it is necessary to conduct a study on the choice of its architecture – the number of input variables, neurons of hidden and context layers, the type of activation functions, and so on.

The results of the experiments showed that the Elman neural network demonstrates a higher accuracy in predicting the time series of agricultural products in comparison with the Jordan network. Further research is suggested towards training recurrent neural networks with different sets of parameters and expand the dataset, as well as combining Elman and Jordan architectures.

References

1. Wei, W. W. S. (2006). *Time series analysis: univariate and multivariate methods* (2nd ed.). Pearson Addison Wesley.

2. Wang, L., Feng, J., Sui, X., Chu, X. & Mu, W. (2020). Agricultural product price forecasting methods: research advances and trend. *British Food Journal*, *122*(7), 2121-2138. https://doi.org/10.1108/BFJ-09-2019-0683

3. Nehrey, M., Kaminskyi, A., & Komar, M. (2019). Agro-economic models: a review and directions for research. *Periodicals of Engineering and Natural Sciences*, *7*(2), 702-711. http://dx.doi.org/10.21533/pen.v7i2.579

4. Zhang, D., Chen, S., Liwen, L., & Xia, Q. (2020). Forecasting agricultural commodity prices using model selection framework with time series features and forecast horizons. *IEEE Access*, *8*, 28197-28209. http://doi.org/10.1109/ACCESS.2020.2971591

5. Kozlovskyi, S., Mazur, H., Vdovenko, N., Shepel, T., & Kozlovskyi, V. (2018). Modeling and forecasting the level of state stimulation of agricultural production in Ukraine based on the theory of fuzzy logic. *Montenegrin Journal of Economics*, *14*(3), 37-53. https://doi.org/10.14254/1800-5845/2018.14-3.3

6. Adewole, A.P., Akinwale, A. T., & Akintomide, A. B. (2011). Artificial Neural Network Model for Forecasting Foreign Exchange Rate. *World of Computer Science and Information Technology Journal*, *1*(3), 110-118. https://www.academia.edu/71618631/Artificial_Neural_Network_Model_for_Forecasting_Foreign_Exchange_Rate

7. Osman, H. (2019). *Time Series Prediction Using Neural Network* (1st ed.). LAP LAMBERT Academic Publishing.

8. Gately, E. (1995). *Neural Networks for Financial Forecasting*. John Wiley & Sons.

9. Palmer, A., Montaño, J.J., & Sesé, A. (2006). Designing an artificial neural network for forecasting tourism time series. *Tourism Management*, *27*(5), 781-790. https://doi.org/10.1016/j.tourman.2005.05.006

10. Lewis, N.D. (2017). *Neural Networks for Time Series Forecasting with R: An Intuitive Step by Step Blueprint for Beginners.* CreateSpace Independent Publishing Platform.

11. Marhon, S.A., Cameron, C.J.F., & Kremer, S.C. (2013). Recurrent Neural Networks. In M. Bianchini, M. Maggini, & L. Jain (Eds.), *Intelligent Systems Reference Library: Vol. 49. Handbook on Neural Information Processing* (pp. 29-65). Springer. https://doi.org/10.1007/978-3-642-36657-4_2

12. Du, K.-L., & Swamy, M.N.S. (2014). Recurrent Neural Networks. In *Neural Networks and Statistical Learning* (pp. 337–353). Springer. https://doi.org/10.1007/978-1-4471-5571-3_11

13. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* MIT Press. https://www.deeplearningbook.org/

14. Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, *14*(2), 179–211. https://doi.org/10.1016/0364-0213(90)90002-E

15. Wang, J., Wang, J., Fang, W., & Niu, H. (2016). Financial Time Series Prediction Using Elman Recurrent Random Neural Networks. *Computational Intelligence and Neuroscience*, *2016*, Article 4742515. https://doi.org/10.1155/2016/4742515

16. Li, C., Zhu, L., He, Z., Gao, H., Yang, Y., Yao, D., & Qu, X. (2019). Runoff Prediction Method Based on Adaptive Elman Neural Network. *Water*, *11*(6), Article 1113. https://doi.org/10.3390/w11061113

17. Ren, G., Cao, Y., Wen, S., Huang, T., & Zeng, Z. (2018). A Modified Elman Neural Network with a New Learning Rate Scheme. *Neurocomputing*, *286*, 11-18. https://doi.org/10.1016/j.neucom.2018.01.046

18. Wysocki, A, & Ławryńczuk, M. (2015). Jordan neural network for modelling and predictive control of dynamic systems. In *Proceedings of 2015 20th International Conference on Methods and Models in Automation and Robotics* (pp. 145-150). IEEE. https://doi.org/10.1109/MMAR.2015.7283862

19. Jordan, M. I. (1997). Serial order: A parallel distributed processing approach. In J. W. Donahoe, & V. P. Dorsel (Eds.), *Advances in Psychology: Vol. 121. Neural-network models of cognition: Biobehavioral foundations* (pp. 471–495). Elsevier Science Publishers. https://doi.org/10.1016/S0166-4115(97)80111-2

20. Putri, T. E., Firdaus, A. A., & Sabilla, W. I. (2018). Short-Term Forecasting of Electricity Consumption Revenue on Java-Bali Electricity System using Jordan Recurrent Neural Network. *Journal of Information Systems Engineering and Business Intelligence*, *4*(2), 96–105. https://doi.org/10.20473/jisebi.4.2.96-105

21. Mohamad, R., Skafi, M., & Haidar, A. (2014). Predicting Global Solar Radiation Using Recurrent Neural Networks and Climatological Parameters. *International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering*, *8*(2), 331-334. https://publications.waset.org/9997510/pdf

22. Shcherbakov, M.V., Brebels, A., Shcherbakova, N.L., Tyukov, A.P., Janosky, T.A., & Kamaev, V.A. (2013). A Survey of Forecast Error Measures. *World Applied Sciences Journal*, *24*, 171-176. https://doi.org/10.5829/idosi.wasj.2013.24.itmies.80032

23. Spiegelhalter, D. (2019). *The Art of Statistics: How to Learn from Data*. Basic Books.

24. Official statistics of Latvia. (2005-2021). *Average retail prices of selected commodity (euro per 1 kg, if other - specified) 2005M01 - 2021M07* [Data set]. Retrieved August 11, 2021, from https://data.stat.gov.lv/pxweb/en/OSP_PUB/START__VEK__PC__PCC/PCC010m/

25. Bergmeir, C., & Benítez, J. (2012). Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS. *Journal of Statistical Software*, *46*(7), 1–26. https://doi.org/10.18637/jss.v046.i07

The article was submitted on 2021, August 23