Long-Term Forecasting Method in the Supply Chain Based on an Artificial Neural Network with Multi-Agent Metaheuristic Training

Eugene Fedorov[a] and Olga Nechyporenko[a]

[a]*Cherkasy State Technological University, Shevchenko blvd., 460, Cherkasy, 18006, Ukraine*

Abstract

The problem of increasing the efficiency of long-term forecasting in the supply chain is examined. Neural network forecasting methods that are based on reservoir calculations, which increases the forecast accuracy, are proposed. Methods for identifying parameters of forecast models based on the metaheuristics are proposed for the methods mentioned above. These methods were researched on the basis of the data from the logistics company Ekol Ukraine and are intended for intelligent computer-based supply chain management systems.

Keywords 1

long-term forecast, supply chain, metaheuristics, reservoir computing, forecast neural network model

1. Introduction

These days, domestic and foreign companies are striving to improve and optimize their business processes with the implementation of the Lean Production technology and principles, the uniqueness of which lies in the fact that it is effective for enterprises of various industries at any stage of the supply chain of products or services to the end consumer. [1-3]. The Lean Production concept is dominant in the formation of "perfect" supply chains, which, in the context of globalization and digitalization of the economy based on information and communication technologies, is the most important factor in competitiveness [4]. One of the most important problems in the field of supply chain management is the insufficiently high accuracy of the forecast. This leads to the fact that supply chain management can be ineffective. Therefore, the development of forecasting methods in the supply chain is an urgent task.

To date, many approaches are known as long-term forecasting tools, among which are:

- autoregressive forecasting methods [5];
- forecasting methods based on exponential smoothing [6];
- neural network forecasting methods [7-10].

Autoregressive methods have the complex determination of the functional dependencies type, the labor intense determination of the model parameters, low adaptability and the lack of the ability to model nonlinear processes.

Neural network forecasting methods provide a tangible advantage, consisting of: the relationships between factors are investigated on ready-made models; no assumptions about the distribution of factors are required; a priori information about factors may be missing; the original data may be highly correlated, incomplete or noisy; analysis of systems with a high degree of nonlinearity is possible; rapid model development; high adaptability; analysis of systems with a large number of

CMIS-2021: The Fourth International Workshop on Computer Modeling and Intelligent Systems, April 27, 2021, Zaporizhzhia, Ukraine
EMAIL: fedorovee@ukr.net (E. Fedorov); olne@ukr.net (O. Nechyporenko)
ORCID: 0000-0003-3841-7373 (E. Fedorov); 0000-0002-3954-3796 (O. Nechyporenko)

factors is possible; a complete enumeration of all possible models is not required; analysis of systems with heterogeneous factors is possible.

However, neural network methods have a lack of transparency, the complexity of the architecture definition, strict requirements for the training sample, the complexity of the training algorithm choice, and the resource-intensiveness of the training process. Therefore, the task of increasing the efficiency of neural network forecast is urgent.

The aim of the work is to develop a method for long-term forecasting in the supply chain. To achieve the goal, the following tasks were set and solved:

- analyze existing forecast methods;
- propose a neural network forecast model;
- choose a criterion for evaluating the effectiveness of a neural network forecast model;
- propose a method for determining the values of the neural network forecast model parameters based on multi-agent metaheuristics;
- perform numerical studies.

2. Problem statement

The problem of increasing the efficiency of long-term forecasting in the supply chain is reduced to the problem of finding such a vector of parameters W, which satisfies the forecast model adequacy criterion $F = \frac{1}{P}\sum_{\mu=1}^{P}(f(x_\mu, W) - d_\mu)^2 \to \min_W$, i.e. deliver the minimum of the mean squared error (the difference between the model output and the desired output), where P – test set cardinality, x_μ .– μ^{th} training input value, d_μ .– μ^{th} training input value.

3. Literature review

The most commonly used forecast neural networks are:

1. Long short-term memory (LSTM) [11, 12];

This network is based on gates (FIR filters) and a multilayer perceptron. Instead of each hidden neuron, it uses a memory block that contains one or more cells, and is connected with input, output and forget gates. Gates determine how much information to pass through. If the input and output gates are close to 1 and the forget gate is close to 0, then the network turns into an Elman network. If the input gate is close to 0, then the short-term information from the input is ignored. If the forget gate is close to 0, then long-term information from the memory block is ignored. If the output gate is close to 0, then the output information is ignored. The advantage of this network is a higher forecast accuracy than in a conventional multilayer perceptron. The disadvantages are the complexity of the memory blocks implementation, insufficient forecast accuracy, the complexity of defining the architecture, insufficient learning rate.

2. Gated recurrent unit (GRU) [13-15];

This network is based on gates (FIR filters) and a multilayer perceptron. Instead of each hidden neuron, it uses a hidden block that is connected with reset and update gates. Gates determine how much information to pass through. If the reset gate is close to 1 and the update gate is close to 0, then the network turns into an Elman's network. If the reset gate and update gate are close to 0, then the long-term information from the hidden block is ignored and the network becomes a multilayer perceptron. If the update gate is close to 1, then the short-term information from the network input is ignored. The advantage of this network is a higher forecast accuracy than in a conventional multilayer perceptron. The disadvantages are the complexity of the hidden blocks implementation, insufficient forecast accuracy, the complexity of defining the architecture, insufficient learning rate.

3. Neural Turing machine (NTM) [16, 17];

This network is based on a Turing machine and a multilayer perceptron or LSTM and includes a controller and a memory matrix. At any given time, the controller receives input from the outside

world and sends the output to the outside world. The controller also reads from the memory matrix cells via the read heads and writes to the memory matrix cells via the write heads. The advantage of this network is a higher forecast accuracy than in a conventional multilayer perceptron. The disadvantages are the complexity of the controller implementation (in the case of LSTM) and the complexity of defining the architecture, insufficient forecast accuracy, insufficient learning rate.

4. Echo state network (ESN) [18, 19];

This network is based on reservoir computing over sigmoid neurons and a multilayer perceptron. The hidden layer is called the reservoir. Each neuron in the reservoir may be unconnected or connected to other neurons in the reservoir. To train the network, the pseudoinverse matrix method is used. The advantages of this network are the highest forecast accuracy (due to the pseudoinverse matrix method) and the ease of implementation of sigmoid neurons in the reservoir. The disadvantages are the complexity of parallel learning and the complexity of defining the architecture.

5. Long short-term memory (LSM) [20-23].

This network is based on reservoir computations over impulse neurons «Leaky Integrate and Fire» (LIF) and multilayer perceptron. Each neuron in the reservoir may be unconnected or connected to other neurons in the reservoir and is excitatory or inhibitory. A gradient learning method is used to train the network. The advantages of this network are a higher forecast accuracy than in a conventional multilayer perceptron and the possibility of parallel training for the part of the network corresponding to a multilayer perceptron. The disadvantages are the complexity of the implementation of impulse neurons, the complexity of defining the architecture and less high prediction accuracy, the complexity of parallel training for the part of the network corresponding to the reservoir.

Usually, the methods listed above either have a low forecast accuracy (due to falling into a local extremum) or a low learning rate (due to the high computational complexity of the hidden neuron or the complexity of parallelization of training) or the complexity of implementation (due to the complexity of the hidden neuron architecture) or the complexity of defining the architecture, which leads to a decrease in forecast efficiency.

Due to this, creation of a neural network with a training method and architecture that will eliminate the indicated disadvantages is an urgent task.

4. Block diagram of a neural network model for a long-term forecast

Figures 1-2 show a block diagram of a long-term forecast model based on a fully connected echo state network (FC-ESN), which is a recurrent two-layer neural network. Unlike traditional ESN, this network is fully connected, using cascades of unit delays. FC-ESN type 1 has a unit delay stage in the input layer. FC-ESN type 2 has a unit delay stage in the input and output layers. The number of input and output neurons is 1.

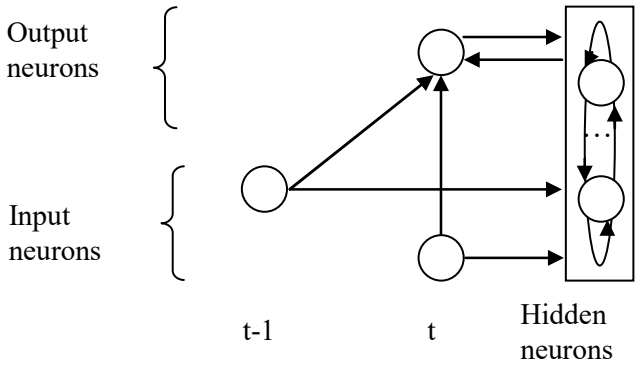

Figure 1: Block diagram of a long-term forecast model based on a fully connected echo state network with a cascade of unit delays for an input layer neuron (FC-ESN type 1)

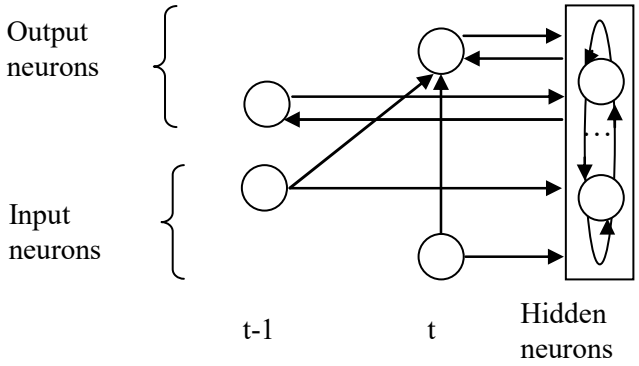

Figure 2: Block diagram of a long-term forecast model based on a fully connected echo-state network with a cascade of unit delays for a neuron of the input and output layers (FC-ESN type 2)

5. Neural network models for long-term forecast

5.1. Long-term forecast model FC-ESN type 1

1. Initialization

$$n = 1 .$$

$$y_i^{(1)}(n-1) = 0 ,\ i \in \overline{1, N^{(1)}} .$$

2. Forecast

2.1. Initialization of the outputs of the neurons of the input layer

$$y_i^{(0)}(n) = x_i ,$$

2.2. Calculation of the outputs of the neurons of the hidden layer

$$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n)) ,\ j \in \overline{1, N^{(1)}} ,$$

$$s_j^{(1)}(n) = b_j^{(1)}(n) + \sum_{i=0}^{M^{(0)}} w_{ij}^{(1)} y^{(0)}(n-i) + \sum_{i=M^{(0)}+1}^{M^{(0)}+N^{(1)}} w_{ij}^{(1)} y_{i-M^{(0)}}^{(1)}(n-1) +$$

$$+ w_{M^{(0)}+N^{(1)}+1}^{(1)} y^{(2)}(n-1) ,$$

2.3. Calculation of the outputs of the neurons of the output layer

$$y^{(2)}(n) = f^{(2)}(s^{(2)}(n)) ,$$

$$s^{(2)}(n) = b^{(2)}(n) + \sum_{i=0}^{M^{(0)}} w_i^{(2)} y^{(0)}(n-i) + \sum_{i=M^{(0)}+1}^{M^{(0)}+N^{(1)}} w_i^{(2)} y_{i-M^{(0)}}^{(1)}(n) .$$

where $N^{(1)}$ – the number of neurons in the first layer,

$M^{(k)}$ – the number of unit delays for the k^{th} layer,

$w_{ij}^{(k)}$ – the connection weight from the i^{th} neuron to the j^{th} neuron on the k^{th} layer,

$b_j^{(k)}$ – displacement (thresholds) on the k^{th} layer,

$y_j^{(k)}(n)$ – the output of the j^{th} neuron on the k^{th} layer at time n,

$f^{(k)}$ – neurons activation function on the k^{th} layer (usually $f^{(k)}(s) = \tanh(s)$).

5.2. Long-term forecast model FC-ESN type 2

1. Initialization

$$n = 1 .$$

$$y_i^{(1)}(n-1) = 0 ,\ i \in \overline{1, N^{(1)}} .$$

2. Forecast

2.1. Initialization of the outputs of the neurons of the input layer

$$y_i^{(0)}(n) = x_i,$$

2.2. Calculation of the outputs of the neurons of the hidden layer

$$y_j^{(1)}(n) = f^{(1)}(s_j^{(1)}(n)),\ j \in \overline{1, N^{(1)}},$$

$$s_j^{(1)}(n) = b_j^{(1)}(n) + \sum_{i=0}^{M^{(0)}} w_{ij}^{(1)} y^{(0)}(n-i) + \sum_{i=M^{(0)}+1}^{M^{(0)}+N^{(1)}} w_{ij}^{(1)} y_{i-M^{(0)}}^{(1)}(n-1) +$$

$$+ \sum_{i=M^{(0)}+N^{(1)}+1}^{M^{(0)}+N^{(1)}+M^{(2)}} w_{ij}^{(1)} y^{(2)}(n-(i-(M^{(0)}+N^{(1)}))),$$

2.3. Calculation of the outputs of the neurons of the output layer

$$y^{(2)}(n) = f^{(2)}(s^{(2)}(n)),$$

$$s^{(2)}(n) = b^{(2)}(n) + \sum_{i=0}^{M^{(0)}} w_i^{(2)} y^{(0)}(n-i) + \sum_{i=M^{(0)}+1}^{M^{(0)}+N^{(1)}} w_i^{(2)} y_{i-M^{(0)}}^{(1)}(n).$$

6. Criterion for evaluating the effectiveness of a neural network model for long-term forecast

In this work, to determine the parameters values of the FC-ESN model, the criterion of the model adequacy was chosen, which means the choice of such values of the parameters $W = \{w_{ij}^{(1)},\ w_i^{(2)}\}$, which deliver the minimum of the mean squared error (the difference between the model output and the desired output):

$$F = \frac{1}{P}\sum_{\mu=1}^{P}(y_\mu^{(2)} - d_\mu)^2 \to \min_W, \qquad (1)$$

where P – the test set cardinality.

7. Method for determining the parameters values of the neural network model for long-term forecast

The method for determining the parameters values of the neural network model for long-term forecasting is reduced to calculating the weights of the hidden layer and the output layer of the FC-ESN model.

7.1. Calculating the weights of the hidden layer

The weights of the hidden layer are calculated as follows:

1. Initialize randomly biases (thresholds) $b_j^{(1)}$ and weights $w_{ij}^{(1)}$.

2. Make up from weights $w_{ij}^{(1)}$, $i \in \overline{M^{(0)}+1, M^{(0)}+N^{(1)}}$, $j \in \overline{1, N^{(1)}}$, matrix $W = [w_{ij}]$, $i, j \in \overline{1, N^{(1)}}$.

3. Determine the matrix $\widehat{W}$ as $\widehat{W} = \alpha \frac{W}{\max\limits_{j \in 1, N^{(1)}} \{|\lambda_j|\}}$,

where α – spectral radius of the matrix $\widehat{W}$ (for large α learning is faster, but long short-term memory decreases), $0 < \alpha < 1$,

λ_j – eigenvalues of matrix W.

4. Assign to the weights $w_{ij}^{(1)}(n)$, $i \in \overline{M^{(0)}+1, M^{(0)}+N^{(1)}}$, $j \in \overline{1, N^{(1)}}$, the values of the corresponding elements of the matrix $\widehat{W}$.

7.2. The output layer weights calculation based on the multi-agent metaheuristic SAPSO method

The proposed SAPSO (simulated annealing and particle swarm optimization) method for numerical functions optimization consists of the following blocks (Figure 3).

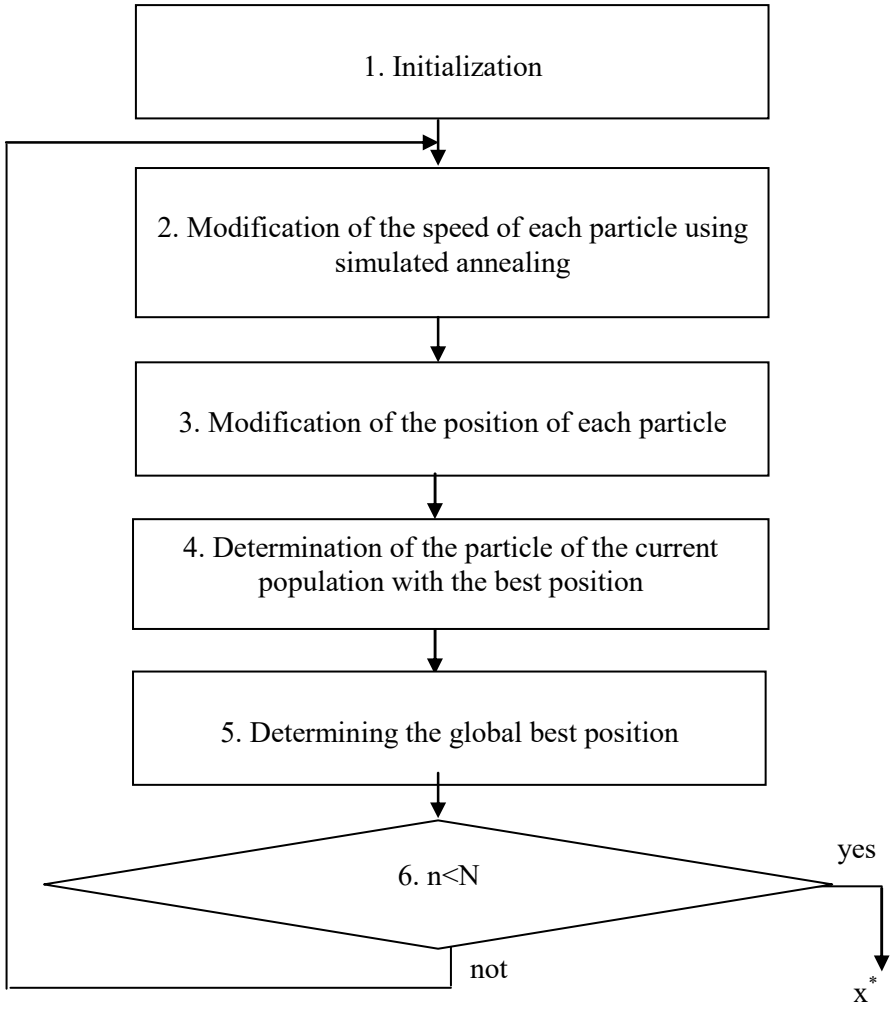

Figure 3: The sequence of procedures of the optimization method based on the multi-agent metaheuristic SAPSO method

Block 1 - Initialization:

- setting the maximum number of iterations N;
- setting the size of the swarm K (usually no more than 40);
- setting the dimension of the particle position M (corresponds to the number of weights in the output layer);
- setting the number of the current iteration n to one;
- initialization of position x_k (corresponds to the solution, i.e. the vector of the weights of the output layer)

$$x_k = (x_{k1},...,x_{kM}),\ x_{ij} = (x_j^{\max} - x_j^{\min})U(0,1) + x_j^{\min},\ k \in \overline{1,K},$$

where $U(0,1)$ – a function that returns a uniformly distributed random number in a range $[0,1]$,

$x_j^{\min}$, $x_j^{\max}$ – minimum and maximum value;

- initialization of personal (local) best position x_k^{best}

$$x_k^{best} = x_k,\ k \in \overline{1,K};$$

- speed initialization v_k

$$v_k = (v_{k1},...,v_{kM}),\ v_{ij} = 0,\ k \in \overline{1,K};$$

- creating an initial particle swarm

$$Q = \{(x_k, x_k^{best}, v_k)\};$$

- determination of the particle of the current population with the best position

$$k^* = \arg\min_{k \in 1,K} F(x_k),$$

$$x^* = x_{k^*}.$$

Block 2 - Modification of the speed of each particle using simulated annealing

Block 2.1 – Calculating two vectors of random numbers for each particle

$$r1_k = (r1_{k1},...,r1_{kM}),\ r1_{kj} \in \{U(0,1), C(0,1), N(0,1)\},\ k \in \overline{1,K},\ j \in \overline{1,M},$$

$$r2_k = (r2_{k1},...,r2_{kM}),\ r2_{kj} \in \{U(0,1), C(0,1), N(0,1)\},\ k \in \overline{1,K},\ j \in \overline{1,M},$$

where $N(0,1)$ – a function that returns a random number from a standard normal distribution,

$C(0,1)$ – a function that returns a random number from a standard Cauchy distribution,

Block 2.2 – Calculating annealing temperature

$$T(n) = \beta T(n-1),\ T(0) = T_0,$$

$$\beta = N^{-\frac{1}{N-1}},\ T_0 = N^{\frac{N}{N-1}},$$

where $T(n)$ – annealing temperature at iteration n,

T_0 – initial annealing temperature,

β – parameter controlling annealing temperature.

Block 2.3 – Calculating parameter controlling the contribution of the component

$$\alpha_1(n) = \alpha_2(n) = \alpha(0)\exp(-1/T(n)),\ w(n) = w(0)\exp(-1/T(n)),$$

$$\alpha(0) = \alpha_0 = 0.5 + \ln 2,\ w(0) = w_0 = \frac{1}{2\ln 2},$$

where $\alpha_1(n)$ – parameter controlling the contribution of the component $(x_k^{best} - x_k)(r_1)^T$ to the particle velocity at the iteration n,

$\alpha_2(n)$ – parameter controlling the contribution of the component $(x^* - x_k)(r_2)^T$ to the particle velocity at the iteration n,

$w(n)$ – parameter controlling the contribution of the particle velocity at iteration n-1 to the particle velocity at iteration n,

α_0 – initial value of parameters $\alpha_1(n)$ and $\alpha_2(n)$,

w_0 – initial value of parameter $w(n)$,

The simulated annealing introduced in this work makes it possible to establish an inverse relationship between parameters $\alpha_1(n)$, $\alpha_2(n)$, $w(n)$ and the iteration number, i.e. at the initial iterations, the entire search space is explored (in this case, the Cauchy distribution is used), and at the final iterations, the search becomes directional (in this case, the normal distribution is used). In addition, in this work, a direct relationship was established between parameters T_0 and β and the iteration number, which makes it possible to automate the selection of these parameters.

The choice of initial values $\alpha_0 = 0.5 + \ln 2$ and $w_0 = \frac{1}{2\ln 2}$ is standard and satisfies the conditions for the particle swarm convergence $w < 1$ and $w_0 > \frac{1}{2}(\alpha_1 + \alpha_2) - 1$.

Block 2.4 – Вычисление speed of each particle

$$v_k = w(n)v_k + \alpha_1(n)(x_k^{best} - x_k)(r_1)^T + \alpha_2(n)(x^* - x_k)(r_2)^T,\ k \in \overline{1,K},$$

Block 3 – Modification of the position of each particle

Block 3.1 Limiting the speed of each particle

$$v_{kj} = \begin{cases} v_{kj} & v_{kj} \in (x_j^{\min}, x_j^{\max}) \\ 0, & v_{kj} \in \{x_j^{\min}, x_j^{\max}\} \end{cases},\ k \in \overline{1,K},\ j \in \overline{1,M}.$$

Block 3.2 – Calculating position of each particle

$$x_k = x_k + v_k,\ k \in \overline{1,K},$$

$$x_{kj} = \begin{cases} x_j^{\min}, & x_{kj} \le x_j^{\min} \\ x_{kj}, & x_{kj} \in (x_j^{\min}, x_j^{\max}) \\ x_j^{\max}, & x_{kj} \ge x_j^{\max} \end{cases},\ k \in \overline{1,K},\ j \in \overline{1,M},$$

Block 4 - Determination of the personal (local) best position of each particle

$$\text{If } F(x_k) \le F(x_k^{best}) \text{, then } x_k^{best} = x_k,\ k \in \overline{1,K}.$$

Block 5 - Determination of the particle of the current population with the best position

$$k^* = \arg\min_{k \in \overline{1,K}} F(x_k).$$

Block 6 - Determining the global best position

$$\text{If } F(x_{k^*}) < F(x^*) \text{, то } x^* = x_{k^*}.$$

Block 7 - Stop condition

If $n < N$, then increase the iteration number n by one and go to block 2.

8. Experiments and results

Modeling of the process of the neural network model values determination was carried out in the Matlab package using Parallel Computing Toolbox. Since the formation of each particle in block 1, the modification of the speed, position and local best position of each particle in blocks 2-4, respectively, occurs independently of other particles, and the order of formation and modification of particles is arbitrary, it is proposed to perform parallel processing of particles using a parallel parfor loop. Parfor is part of Parallel Computing Toolbox, replaces the sequential for loop and is based on OpenMP technology, but unlike it, it can be used not only on a local multicore machine, but also on a cluster. The advantage of this approach over the CUDA and MPI technologies (represented in the Parallel Computing Toolbox by the spmd block) is the simplicity and clarity of the technical implementation. Due to the small number of particles, it becomes possible to perform the formation and modification of each particle on the corresponding physical core of the machines processors united in a cluster.

Swarm size was selected as K=40.

To determine the type of distribution used in the SAPSO method, a number of experiments were carried out, the results of which are presented in Table 1.

Table 1

Comparative characteristics of distribution types

Distribution type / Criterion	U(0,1)	N(0,1)	C(0,1)
Number of iterations	1000	10000	100000

According to Table 1, the distribution U(0,1) requires the least number of iterations while maintaining the required forecast accuracy.

To define the structure of a long-term forecast model based on FC-ESN, i.e. determining the number of hidden neurons, a number of experiments were carried out, the results of which are presented in Figure 4.

A sample of values based on data from the logistics company Ekol Ukraine was used as input data to determine the parameters values of the neural network model for the long-term forecast. The criterion for choosing the structure of the neural network model was the minimum mean squared forecast error. As can be seen from Figure 4, with an increase in the number of hidden neurons, the error value decreases. It is sufficient to use 16 neurons in the hidden layer for the forecast, since with a further increase in the number of neurons in the hidden layer, the change in the error value is insignificant.

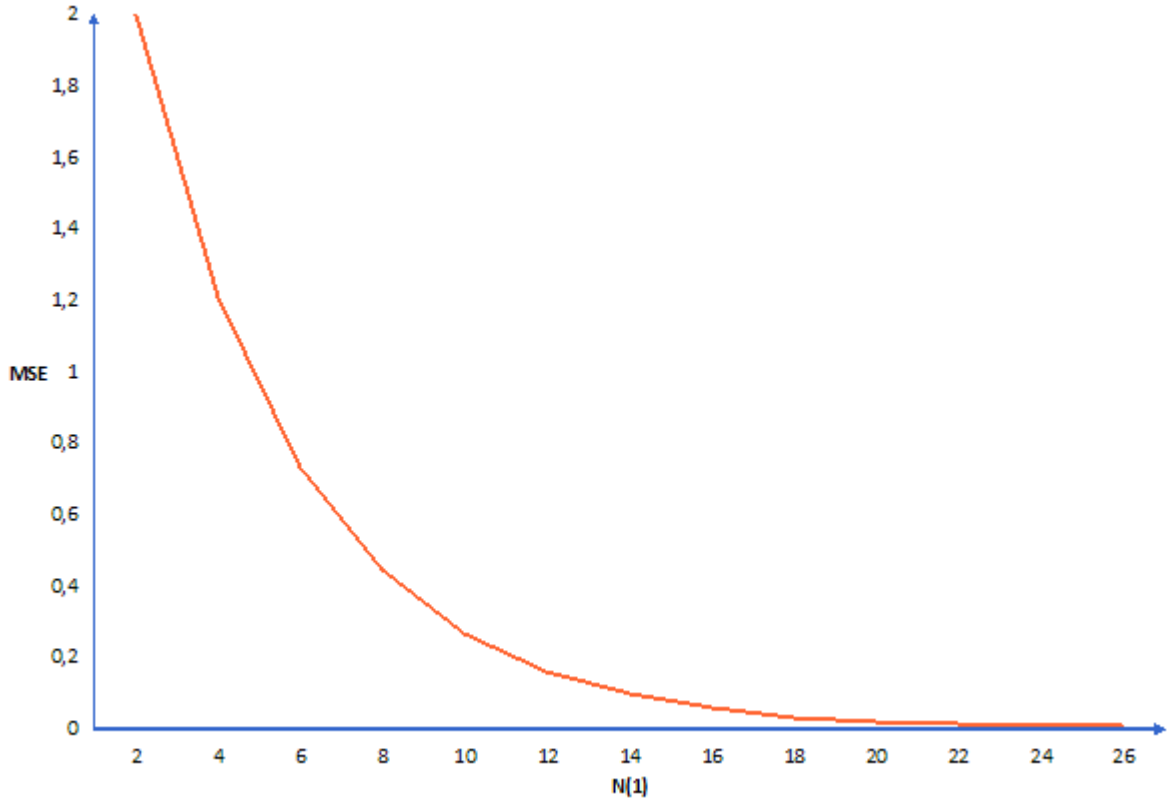

Figure 4: Graph of the dependence of the mean square error (MSE) of the forecast on the number of hidden neurons

The neural networks for long-term forecasting were investigated in the work according to the criterion of the minimum mean squared error (MSE) of the forecast and computational complexity (Table 2), where $M^{(k)}$ – the number of unit delays for the k^{th} layer, S – the number of cell, $N^{(1)}$ – the number of neurons in the first layer, P – training set cardinality, N – number of iterations of the multi-agent metaheuristic method SAPSO, $N << P$, $N^{(1)} << P$.

According to Table 2, FC-ESN type 2 has the highest forecast accuracy, and Type 1 FC-ESN network has the lowest computational complexity.

Based on the performed experiments, the following conclusions can be drawn.

The LSTM network has average learning rates and forecast accuracy.

Table 2
Comparative characteristics of neural networks for long-term forecast

Network / Criterion	Full LSTM	GRU	ESN	FC-ESN type 1/2
Minimum MSE of the forecast	0.12	0.20	0.08	0.06 / 0.02
Computational complexity	$\sim PN^{(1)}(5M^{(0)}+ 3M^{(0)}S+24S+S^2)$	$\sim PN^{(1)}6 (M^{(0)}+N^{(1)})$	$\sim PN^{(1)}(M^{(0)}+ N^{(1)})+(\max\{P,M^{(0)}+N^{(1)}\})^2$	$\sim N(M^{(0)}+N^{(1)})$ / $\sim N(M^{(0)}+M^{(2)}+ N^{(1)})$

The GRU network is second only to the author's networks in learning speed (it uses a gradient learning method and less computational complexity than LSTM and ESN). But it has the least prediction accuracy (due to the gradient learning method and a simplified architecture compared to LSTM).

ESN networks are inferior in forecast accuracy only to the author's networks, since they are trained on the basis of the pseudoinverse matrix method. But it has the lowest learning rate (it has the highest computational complexity, and the pseudoinverse matrix method does not provide for parallelism).

The author's FC-ESN networks are trained on the basis of the proposed metaheuristic, which increases the forecast accuracy (low probability of hitting the local extremum) and the learning rate (provides parallel learning), and does not have the complex implementation.

9. Conclusions

The article discusses the problem of improving the efficiency of long-term forecasting in the supply chain. To solve this problem, the existing forecasting methods were investigated. These studies have shown that by far the most effective is the use of artificial neural networks. To improve the quality of the long-term forecast, an ESN neural network was chosen, modified (by introducing full connectivity and cascades of unit delays in the input and output layers), and in the course of a numerical study, the structure of its model was determined. The experiments have shown that with 16 hidden neurons, the value of the mean squared error does not change significantly, and the selected network gives forecast results with a minimum deviation. A method was proposed for determining the parameters values of the proposed neural network model for long-term forecast. This allowed to ensure high speed and accuracy of the forecast. The proposed methods are intended for software implementation in the Matlab package using Parallel Computing Toolbox, which speeds up the process of finding a solution. The software implementing the proposed methods was developed and researched on the database of the logistics company Ekol Ukraine. The conducted experiments have confirmed the efficiency of the developed software allowing to recommend it for practical use in solving problems of supply chain management. Prospects for further research are in applying the proposed methods on a wider set of benchmarks.

10.References

[1] J. F. Cox, J. G. Schleher, Theory of constraints handbook, New York, NY, McGraw-Hill, 2010.

[2] E. M. Goldratt, My saga to improve production, in: Selected Readings in Constraints Management, Falls Church, VA: APICS, 1996, pp. 43-48.

[3] E. M. Goldratt, Production: The TOC way, including CD-ROM simulator and workbook, Revised edition, Great Barrington, MA: North River Press, 2003.

[4] S. Smerichevska et al, Cluster Policy of Innovative Development of the National Economy: Integration and Infrastructure Aspects: monograph, S. Smerichevska (Eds.), Wydawnictwo naukowe WSPIA, 2020.

[5] R. T. Baillie, G. Kapetanios, F. Papailias, Modified information criteria and selection of long memory time series models, in: Computational Statistics and Data Analysis, volume 76, 2014, pp. 116–131. doi: 10.1016/j.csda.2013.04.012.

[6] P. Bidyuk, T. Prosyankina-Zharova, O. Terentiev, Modelling nonlinear nonsta-tionary processes in macroeconomy and finances, in: Z. Hu, S. Petoukhov, I. Dychka, M. He (Eds.), Advances in Computer Science for Engineering and Education. Advances in Intelligent Systems and Computing, volume 754, Springer, Cham, 2019, pp. 735–745. doi: 10.1007/978-3-319-91008-6_72.
[7] L. Lyubchyk, E. Bodyansky, A. Rivtis, Adaptive harmonic components detection and forecasting in wave non-periodic time series using neural networks, in: Proceedings of the ISCDMCI'2002, Evpatoria, 2002, pp. 433-435.
[8] K.-L. Du, K. M. S. Swamy, Neural networks and statistical learning, Springer-Verlag, London, 2014.
[9] S. Haykin, Neural networks, Pearson Education, New York, NY, 1999.
[10] S. N. Sivanandam, S. Sumathi, S. N. Deepa, Introduction to neural networks using Matlab 6.0, The McGraw-Hill Comp., Inc., New Delhi, 2006.
[11] S. Hochreiter, J. Schmidhuber, Long short-term memory, in: Neural Computation, volune 9, 1997, pp. 1735-1780. doi: 10.1162/neco.1997.9.8.1735.
[12] F. Gers, Long Short-Term Memory in Recurrent Neural Networks, PhD thesis, Ecole Polytechnique Federale de Lausanne.
[13] K. Cho, B. van Merrienboer, C. Gulcehre, F Bougares, H Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 2014, pp. 1724–1734. doi: 10.3115/v1/D14-1179.
[14] R. Dey, F. M. Salem, Gate-Variants of Gated Recurrent Unit (GRU) Neural Networks, arXiv:1701.05923, 2017. URL: https://arxiv.org/ftp/arxiv/papers/1701/1701.05923.pdf.
[15] E. Fedorov, T. Utkina, O. Nechyporenko, Forecast method for natural language constructions based on a modified gated recursive block, in: CEUR Workshop Proceedings, vol. 2604, 2020, pp. 199-214.
[16] A. Graves, G. Wayne, M. Reynolds et al., Hybrid computing using a neural network with dynamic external memory, Nature 538 (2016) 471–476. doi:10.1038/nature20101.
[17] R. B. Greve, E. J. Jacobsen, S. Risi, Evolving neural turing machines for reward-based learning, in: Proceedings of the 2016 Genetic and Evolutionary Computation Conference, GECCO'16, ACM, 2016, pp. 117–124. doi: 10.1145/2908812.2908930.
[18] H. Jaeger, Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the Echo State Network Approach, GMD Report 159, German National Research Center for Information Technology, 2002.
[19] H. Jaeger, M. Lukosevicius, D. Popovici, U. Siewert, Optimization and applications of echo state networks with leakyintegrator neurons, in: Neural Networks volume 20, 2007, pp. 335–352. doi:10.1016/j.neunet.2007.04.016.
[20] T. Natshlager, W. Maas, H. Markram, The liquid computer: A novel strategy for real-time computing on time series, in: Special Issue on Foundations of Information Processing of Telematik, 2002, pp. 39–43.
[21] Q. Wang, P. Li, D-lsm: Deep liquid state machine with unsupervised recurrent reservoir tuning, in Pattern Recognition (ICPR), in: 23rd International Conference on Pattern Recognition (ICPR) (Cancun: IEEE), 2016, pp. 2653–2658. doi: 10.1109/ICPR.2016.7900035.
[22] W. Maass, Liquid state machines: motivation, theory, and applications, in: Computability in context: computation and logic in the real world, 2011, pp. 275–296. doi: 10.1142/9781848162778_0008.
[23] T. Neskorodieva, E. Fedorov, I. Izonin, Forecast method for audit data analysis by modified liquid state machine, in: CEUR Workshop Proceedings, 2020, volume 2631, pp. 145-158.