Mathematical modeling

Математическое моделирование

UDC 330.4
https://doi.org/10.32362/2500-316X-2024-12-4-106-116
EDN WDYUFJ

RESEARCH ARTICLE

Neural network analysis in time series forecasting

Bakhtierzhon Pashshoev,
Denis A. Petrusevich @

MIREA — Russian Technological University, Moscow, 119454 Russia
@ *Corresponding author, e-mail: petrusevich@mirea.ru, petrdenis@mail.ru*

Abstract

Objectives. To build neural network models of time series (LSTM, GRU, RNN) and compare the results of forecasting with their mutual help and the results of standard models (ARIMA, ETS), in order to ascertain in which cases a certain group of models should be used.

Methods. The paper provides a review of neural network models and considers the structure of RNN, LSTM, and GRU models. They are used for modeling time series in Russian macroeconomic statistics. The quality of model adjustment to the data and the quality of forecasts are compared experimentally. Neural network and standard models can be used both for the entire series and for its parts (trend and seasonality). When building a forecast for several time intervals in the future, two approaches are considered: building a forecast for the entire interval at once, and step-by-step forecasting. In this way there are several combinations of models that can be used for forecasting. These approaches are analyzed in the computational experiment.

Results. Several experiments have been conducted in which standard (ARIMA, ETS, LOESS) and neural network models (LSTM, GRU, RNN) are built and compared in terms of proximity of the forecast to the series data in the test period.

Conclusions. In the case of seasonal time series, models based on neural networks surpassed the standard ARIMA and ETS models in terms of forecast accuracy for the test period. The single-step forecast is computationally less efficient than the integral forecast for the entire target period. However, it is not possible to accurately indicate which approach is the best in terms of quality for a given series. Combined models (neural networks for trend, ARIMA for seasonality) almost always give good results. When forecasting a non-seasonal heteroskedastic series of share price, the standard approaches (LOESS method and ETS model) showed the best results.

Keywords: dynamic series, macroeconomic statistics, GRU, LSTM, RNN, DNN, time series

• **Submitted:** 21.06.2023 • **Revised:** 15.02.2024 • **Accepted:** 26.05.2024

For citation: Pashshoev B., Petrusevich D.A. Neural network analysis in time series forecasting. *Russ. Technol. J.* 2024;12(4):106–116. https://doi.org/10.32362/2500-316X-2024-12-4-106-116

Financial disclosure: The authors have no a financial or property interest in any material or method mentioned.

The authors declare no conflicts of interest.

НАУЧНАЯ СТАТЬЯ

Анализ нейросетевых моделей для прогнозирования временных рядов

Б. Пашшоев,
Д.А. Петрусевич [@]

МИРЭА — Российский технологический университет, Москва, 119454 Россия
[@] *Автор для переписки, e-mail: petrusevich@mirea.ru, petrdenis@mail.ru*

Резюме

Цели. Основная цель работы – построить нейросетевые модели временных рядов (LSTM, GRU, RNN) и сравнить результаты прогнозирования с их помощью между собой и с результатами стандартных моделей (ARIMA, ETS), чтобы выяснить, в каких случаях следует пользоваться определенной группой моделей.

Методы. Проведен обзор нейросетевых моделей, рассмотрена структура моделей RNN, LSTM, GRU. Они используются для моделирования временных рядов российской макроэкономической статистики. Качество подстройки моделей под данные и качество прогнозов сравниваются в эксперименте. Нейросетевые и стандартные модели могут применяться как для всего ряда целиком, так и для его частей (тренд и сезонность). При построении прогноза на несколько временных промежутков вперед рассматриваются два подхода: построение прогноза сразу на весь промежуток и пошаговый прогноз. Так появляется несколько комбинаций моделей, которые могут использоваться для прогнозирования. Эти подходы проанализированы в вычислительном эксперименте.

Результаты. Проведено несколько экспериментов, в которых построены и сравниваются по близости прогноза к данным ряда в тестовом периоде стандартные (ARIMA, ETS, LOESS) и нейросетевые модели (LSTM, GRU, RNN).

Выводы. Для сезонных временных рядов модели на основе нейронных сетей превзошли по точности прогноза на тестовый период времени стандартные модели ARIMA, ETS. Одношаговый прогноз вычислительно менее эффективен, чем интегральный прогноз на весь целевой период, но точно указать, для каких рядов какой именно подход оказывается лучшим по качеству, не удается. Комбинированные модели (нейронные сети для тренда, ARIMA – для сезонности) почти всегда дают хороший результат. При прогнозировании несезонного гетероскедастичного ряда курса акций лучшие результаты показали стандартные подходы (метод LOESS и модель ETS).

Ключевые слова: динамические ряды, макроэкономическая статистика, GRU, LSTM, RNN, DNN, временные ряды

• **Поступила:** 21.06.2023 • **Доработана:** 15.02.2024 • **Принята к опубликованию:** 26.05.2024

Для цитирования: Пашшоев Б., Петрусевич Д.А. Анализ нейросетевых моделей для прогнозирования временных рядов. *Russ. Technol. J*. 2024;12(4):106–116. https://doi.org/10.32362/2500-316X-2024-12-4-106-116

Прозрачность финансовой деятельности: Авторы не имеют финансовой заинтересованности в представленных материалах или методах.

Авторы заявляют об отсутствии конфликта интересов.

INTRODUCTION

This article analyzes the application of common neural network models for time series forecasting. Much research has been devoted to the topic of time series forecasting. In fact, several off-the-shelf approaches are used in practice, such as: ARIMA (autoregressive integrated moving average) models, ETS (exponential smoothing) models [1, 2], construction of regressions reflecting dependencies between time-varying parameters. These can be referred to statistical models [3]. GARCH (generalized autoregressive conditional heteroskedasticity) models are used when establishing the phenomenon of heteroskedasticity [1, 2]. Ready-made neural network models LSTM (long short-term memory) and GRU (gated recurrent unit) can be trained using available time series data. There are many publications where several models of different types are built at once to describe a certain temporal process and their forecasts are used together (they are specified below in the description of models). Estimation of forecast accuracy when applying a combination of ARIMA models is discussed [4, 5]. Due to the availability of a multitude of models, the question of which of them should be used for modeling the time process depending on its properties becomes essential [6]. The experimental part of the work considers the representation of seasonal monthly time series of personal income (HHI), and the real agricultural production index (AGR).[1] Non-seasonal time series is represented by stock prices and stock indices (in particular, the SberBank stock price).[2] The main objective of the paper is to determine which models should be used for modeling time processes.

The experimental section considers the construction of time series models ARIMA, neural network models LSTM, GRU, recurrent neural networks (RNN), and full-connected neural networks. Their forecasts for the test period are compared. The quality of neural network models built on such data is compared with the quality of ARIMA/ETS statistical models by information criteria and the quality of forecasts for the test period.

CONSIDERED APPROACHES TO TIME SERIES SIMULATION

When forecasting a time series, a model can be built in many ways. In particular, it is possible to train a neural network or build a statistical model based on the initial values of the time series. However, on the other hand, it is possible to use the division of the series into a seasonal component and a trend.

Usually, the trend T_t is a deterministic part of the time series y_t with a seasonal component S_t (it may not exist), and noise R_t, where t is time. The series can be represented in an additive or multiplicative form:

$$y_t = S_t + T_t + R_t, \quad (1)$$

$$y_t = S_t \times T_t \times R_t. \quad (2)$$

These approaches are equivalent.

In this case, one of the most common models for describing a time series that does not rely on neural networks is ARIMA(p, d, q). This consists of the autoregressive part (for a model of order p the values of the series X are made dependent on p of their previous values):

$$X_t = c + \varphi_1 X_{t-1} + \ldots + \varphi_p X_{t-p},$$

where $\varphi_i, i = \overline{1, p}$ are the coefficients of the function; and from the moving average part of the order q [1]:

$$X_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \ldots + \theta_q \varepsilon_{t-q}.$$

where $\theta_i, i = \overline{1, q}$ are the coefficients of the function. The order d denotes the number of differentiations of the series.

In fact, when building a model, the trend is overcome by switching to a stationary time difference (by repeatedly differentiating the series until the statistical test confirms stationarity) [1, 6]. The work is carried out with the transformed stationary time series. As part of the computational part of the study, we compare its results with the forecasts of the other models.

Since, according to decompositions (1), (2), the parts responsible for seasonal fluctuations and noise can be separated during modeling, the neural network can be trained both on the basis of original data and separately on the basis of trend. Because such separation is possible, several approaches to training data for neural network training are presented in the computational experiment. Neural network models enable forecasting both trend and seasonality, so trend and seasonality can be separately predicted using their own models and the results combined. In the second approach the data is not separated (used, for example, in ARIMA, ETS models). When modeling the trend, the time series is first separated into trend, seasonal component and noise. A neural network model is trained on the basis of trend data, except that the trend is predicted after training. Then the final forecast is collected from the trend forecast, as well as the seasonal component and noise models. The

[1] Unified archive of economic and sociological data. Dynamic series of macroeconomic statistics of the Russian Federation. Indices of wages, monetary incomes of the population; real volume of agricultural production. https://web.archive.org/web/20230317111717/http://sophist.hse.ru/hse/nindex.shtml (in Russ.). Accessed June 01, 2024.

[2] SberBank share price (SBER). https://www.moex.com/ru/issue.aspx?board=TQBR&code=SBER (in Russ.). Accessed June 01, 2024.

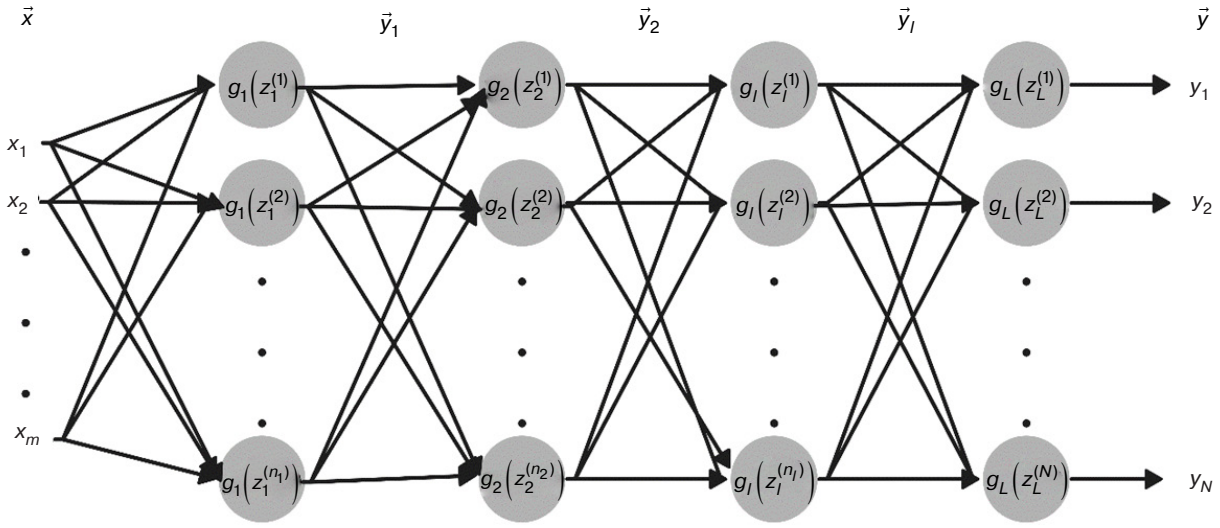

Fig. 1. DNN structure [8]. $\vec{x}$ is the input vector, $\vec{y}_i$ is the vector at the output of the *i*th layer of the network, $\vec{y}$ is the output vector of the entire network (result), $g(.)$ is the activation function, z_i^j is the input of the *j*th neuron in the *i*th layer is the weighted linear combination of the results of the previous layer (weights are adjusted during training)

separation into trend, noise and seasonality is done using LOESS (STL)[3] [1].

In addition, forecasting itself can also be done in two ways. Researchers are usually interested in forecasting not one step ahead, but for several or for the entire season (if the series is seasonal). In this way, it is possible to assess how well the model describes the data of the series. However, the forecast for several steps ahead can be made either at once (integral forecast) or one step at a time (one-step forecast). In the second case, each predicted value becomes a new part of the training sample, on the basis of which the model is constantly adjusted, while the forecast itself is made only one step ahead at each iteration. Both approaches are compared in a computational experiment in the form of single-step and multi-step forecasting.

Several models are involved in the computational experiment: dense neural networks (DNN), recurrent neural networks (RNN), long short-term memory networks (LSTM), and guided recurrent unit (GRU).

Fully connected neural networks are a widely known neural network architecture [7]. Each neuron receives a signal from all neurons of the previous layer (except for the network inputs), applies an activation function to their weighted combination and transmits the result to the neurons of the next layer. Various optimization methods are used to train fully-connected neural networks, such as gradient descent and its modifications. However, due to the large number of parameters, fully-connected networks can be prone to overtraining. Regularization methods such as L1 and L2 and dropout methods are used to combat overtraining. The structure of the network is shown in Fig. 1.

Recurrent neural networks RNN [9, 10] are used to simulate functional relationships between input features in the recent past, and the target variable in the future. As shown in Fig. 2, an RNN is periodically trained on a historical dataset, focusing on internal (hidden) state transitions from time state $t - 1$ to the cutoff t. The resulting model is defined by two weight matrices $\mathbf{W}_{xs}$ and $\mathbf{W}_y$, and two bias vectors $\mathbf{b}_s$ and $\mathbf{b}_y$. The output $\mathbf{y}_t$ depends on the internal state $\mathbf{S}_t$, which depends on both the current input $\mathbf{x}_t$ and the previous state $\mathbf{S}_{t-1}$:

$$\mathbf{S}_t = \text{th}[\mathbf{W}_{xs}(\mathbf{x}_t \oplus \mathbf{S}_{t-1}) + \mathbf{b}_s],$$
$$\mathbf{y}_t = \sigma(\mathbf{W}_y \mathbf{S}_t + \mathbf{b}_y).$$

Here $\mathbf{x}_t$ is the input vector at time t, $\sigma(\mathbf{x})$ is a sigmoid function, and the operation $\oplus$ is a concatenation. The main disadvantage of RNN is the problem of gradient decay, due to which the gradient becomes smaller over time. This is expressed by the fact that RNN memorizes information for only short periods of time.

Long Short Term Memory (LSTM) networks [11–26] are a variant of RNNs which partially resolve the fading gradient problem and learn longer term dependencies in time series. They are described at time t in terms of the internal (hidden) state $\mathbf{S}_t$ and the cell state $\mathbf{C}_t$. The state $\mathbf{C}_t$ depends on three parameters: the previous cell state $\mathbf{C}_{t-1}$; the previous internal state $\mathbf{S}_{t-1}$; and the input at the current time $\mathbf{x}_t$. The process depicted in Fig. 3, enables

[3] LOESS—locally estimated scatterplot smoothing; STL (seasonal and trend decomposition using LOESS)—method of decomposition of time series into trend, seasonality, and residuals.

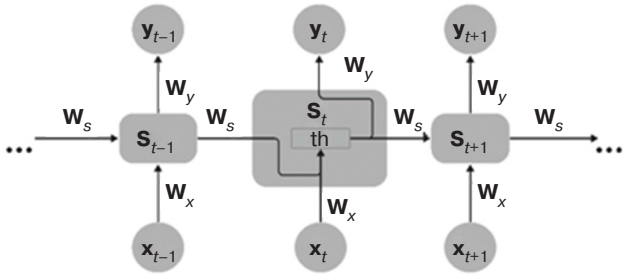

Fig. 2. RNN structure [8]

moving/filtering, multiplying/uniting, and adding information using forget, input, addition, and output gates implemented by the functions $\mathbf{x}_t$, $\mathbf{i}_t$, $\tilde{\mathbf{C}}_t$, and $\mathbf{O}_t$, respectively. This enables more precise control of learning long-term dependencies.

These functions are related as follows:

$$\mathbf{f}_t = \sigma(\mathbf{W}_f(\mathbf{x}_t \oplus \mathbf{S}_{t-1}) + \mathbf{b}_f),$$
$$\mathbf{i}_t = \sigma(\mathbf{W}_i(\mathbf{x}_t \oplus \mathbf{S}_{t-1}) + \mathbf{b}_i),$$
$$\tilde{\mathbf{C}}_t = \text{th}(\mathbf{W}_c(\mathbf{x}_t \oplus \mathbf{S}_{t-1}) + \mathbf{b}_c),$$
$$\mathbf{C}_t = \mathbf{f}_t\mathbf{C}_{t-1} + \mathbf{i}_t\tilde{\mathbf{C}}_t,$$
$$\mathbf{O}_t = \sigma(\mathbf{W}_O(\mathbf{x}_t \oplus \mathbf{S}_{t-1}) + \mathbf{b}_0),$$
$$\mathbf{S}_t = \mathbf{O}_t\text{th}(\mathbf{C}_t),$$
$$\mathbf{y}_t = \sigma(\mathbf{W}_y\mathbf{S}_t + \mathbf{b}_y),$$

where $\mathbf{W}_f$, $\mathbf{W}_i$, $\mathbf{W}_c$, $\mathbf{W}_O$, $\mathbf{W}_y$ are various weight matrices involved in the training. The functions are used for forecasting both independently (e.g., the spread of coronavirus in [12, 19, 26] is modeled based on LSTM) and in combination with other models in [14, 15]. A combination of the forecast of this model with the results of other models can be applied. In [12, 16, 24, 26], deep learning is used to tune LSTM-based models. In [11, 13, 19, 20], LSTM-based models are compared with other commonly used models in forecasting a certain time process.

Guided recurrence units (GRU) [13, 25, 27–29] are a variant of LSTM which can resolve the fading gradient problem even better. As can be seen from Fig. 4, the novelty of this method lies in the use of update, reset, and third gates implemented by functions $\mathbf{z}_t$, $\mathbf{r}_t$, $\tilde{\mathbf{S}}_t$ Each element has a different role in controlling the filtering, utilization, and merging of the previous information. The first term in the expression for the following state $(1-\mathbf{z}_t)\mathbf{S}_{t-1}$ enables configuration of what to keep from the past, while the element $\mathbf{z}_t\tilde{\mathbf{S}}_t$ determines what to use from current memory contents.

These functions are related as follows:

$$\mathbf{r}_t = \sigma(\mathbf{W}_r(\mathbf{x}_t \oplus \mathbf{S}_{t-1}) + \mathbf{b}_r),$$
$$\mathbf{z}_t = \sigma(\mathbf{W}_z(\mathbf{x}_t \oplus \mathbf{S}_{t-1}) + \mathbf{b}_z),$$
$$\tilde{\mathbf{S}}_t = \text{th}(\mathbf{W}_s(\mathbf{x}_t \oplus \mathbf{S}_{t-1}\mathbf{r}_t) + \mathbf{b}_s),$$
$$\mathbf{S}_t = (1-\mathbf{z}_t)\mathbf{S}_{t-1} + \mathbf{z}_t\tilde{\mathbf{S}}_t,$$
$$\mathbf{y}_t = \sigma(\mathbf{W}_y\mathbf{S}_t + \mathbf{b}_y).$$

Both applications of the GRU element connects with other neural networks (in [27, 28] with CNN networks [7]) and cascaded element construction [29] can be found in the literature.

COMPUTATIONAL EXPERIMENT

This work presents the results of three experiments on the representation of monthly time series: household income (HHI); the index of the real agricultural production (AGR) (the indices have dimensionless units); and the daily time series of the SberBank of the Russian Federation share price measured in rubles.

The fully connected neural network has the structure shown in the Table 1.

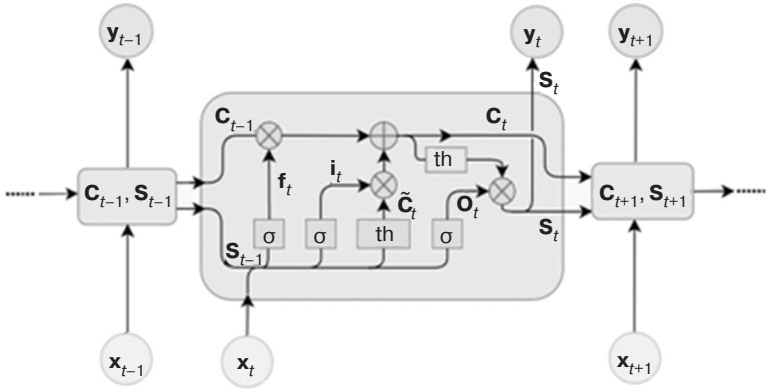

Fig. 3. LSTM structure [8]

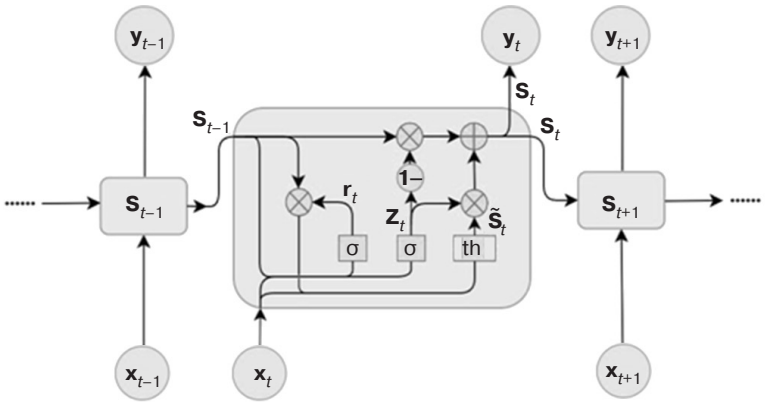

Fig. 4. GRU structure [8]

Table 1. Structure of a fully connected DNN neural network

Forecast type	Layer number	Number of neurons
Single-step	Input layer, 0	6
	Layers 1, 5	64
	Layers 2–4	128
	Output layer, 6	1
Integral (forecast for 12 time periods ahead)	Input layer, 0	24
	Layers 1, 5	64
	Layers 2–4	128
	Output layer, 6	12

In single-step forecasting, the network makes a forecast 1 step ahead. This data then becomes part of the training data, and the next step is made (one neuron in the output layer). In the integral approach, the forecast is made for 12 steps forward at once (for a year, since a series with annual seasonality is considered; there are 12 neurons in the output layer).

In order to evaluate model forecasts, measures of the closeness of the forecast vector and the vector of real values of the quantity are considered [1, 2]:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_t (\tau(t) - ts(t))^2}{N}}, \qquad \mathrm{MAE} = \frac{\sum_t |\tau(t) - ts(t)|}{N}. \tag{3}$$

Here RMSE is a root mean square error; MAE is a mean absolute error; $\tau(t)$ are the real values of the time series; $ts(t)$ is the forecast of the mathematical model; N is the length of the forecasting segment (most often, it coincides with seasonality, and here we are talking about time series with annual seasonality, so $N = 12$).

The RNN network architecture chosen for integral trend forecasting with a step of 12 consists of three RNN layers (each containing 64 neurons), as well as a single layer of fully-connected neurons (i.e., 12 neurons in terms of the number of predicted values). A time window of size **w** is fed to the input of the model. As a result of experiments, it was found that a single layer is unable to detect seasonality. Adding more than three layers does not significantly improve the quality of the forecast. For this reason, three layers are chosen. The total number of trained parameters in the model is 21516.

A simpler model containing only one RNN layer with 64 neurons and an input layer with a single neuron was chosen for single-step trend forecast. This resulted in a significant reduction in the number of trained

parameters to 4289, since additional layers did not noticeably improve the forecast quality.

LSTM and GRU architectures were chosen to be identical in terms of the structure of the RNN network model described above. However, the number of trained parameters of LSTM for the two different architectures is 83724 and 16961, respectively, while for GRU it is 63564 and 12929.

The classical ARIMA and ETS models are involved in the experiments [1, 2]. In the LOESS method, a trend is extracted based on the STL decomposition. It is then forecast for the test period using the ARIMA model. A seasonality model is superimposed on the forecast for the test period. In addition, the trend was estimated using a polynomial. Seasonality was estimated using ARIMA, and the results were combined.

Experiment 1 considers the index of money income of the Russian population for 2000–2020. All the considered models were adapted for the 2000–2020 training period (the crisis years 2008 and 2014 were removed and the data agglomerated). The results of their forecasts for the test period (2021) are compared in Table 2.

Table 2. Money income index models according to macroeconomic statistics of the Russian Federation and their forecasts for the test period

Time series model	MAE	RMSE
Polynomial of degree 4 + seasonality ARIMA(1, 1, 2)	3.42	4.52
LOESS method	3.49	4.57
ARIMA(6, 1, 5) with the seasonality $(0, 1, 1)_{12}$	5.86	7.01
ETS	6.57	8.47
DNN model for trend, single-step forecast	4.21	5.58
DNN model for trend, integral forecast	3.88	4.58
DNN model for trend and seasonality, single-step forecast	2.44	3.06
DNN model for trend, ARIMA(1, 1, 2) for seasonality, single-step forecast	**1.73**	**1.97**
DNN model for trend and seasonality, integral forecast	2.48	3.36
DNN model for trend, ARIMA(1, 1, 2) model for seasonality, integral forecast	**2.29**	**2.62**
RNN model for trend, single-step forecast	6.25	7.68
RNN model for trend, integral forecast	4.65	5.86
RNN model for trend and seasonality, single-step forecast	4.32	4.72
RNN model for trend, ARIMA(1, 1, 2) model for seasonality, single-step forecast	2.82	3.3
RNN model for trend and seasonality, integral forecast	3.88	4.45
RNN model for trend, ARIMA(1, 1, 2) model for seasonality, integral forecast	**2.35**	**2.95**
LSTM model for trend, single-step forecast	23.43	30.68
LSTM model for trend, integral forecast	18.97	30.09
LSTM model for trend and seasonality, single-step forecast	3.83	4.25
LSTM model for trend, ARIMA(1, 1, 2) model for seasonality, single-step forecast	**2.42**	**2.79**
LSTM model for trend and seasonality, integral forecast	5.91	6.63
LSTM model for trend, ARIMA model for seasonality, integral forecast	5.03	5.40
GRU model for trend, single-step forecast	19.00	29.28
GRU model for trend, integral forecast	20.05	27.30
GRU model for trend and seasonality, single-step forecast	3.81	4.24
GRU model for trend, ARIMA(1, 1, 2) model for seasonality, single-step forecast	**2.40**	**2.76**
GRU model for trend and seasonality, integral forecast	3.94	4.36
GRU model for trend, ARIMA(1, 1, 2) model for seasonality, integral forecast	**2.41**	**2.89**

Table 3. Models of the index of real volume of agricultural production according to macroeconomic statistics of the Russian Federation and their forecasts for the test period

Time series model	MAE	RMSE
Polynomial of degree 1 + ARIMA(2, 0, 1) with seasonality $(2, 1, 1)_{12}$	67.04	77.76
Logarithmic function $y = a_0 + a_1 \ln x$	55.04	80.92
Exponential function $y = \exp(a_0 + a_1 x)$	53.00	90.48
ARIMA(3, 0, 1) with a seasonality $(2, 1, 2)_{12}$	13.24	18.51
ETS	17.22	25.40
DNN model for trend, single-step forecast	15.97	29.70
DNN model for trend, integral forecast	14.63	26.61
DNN model for trend and seasonality, single-step forecast	9.75	16.18
DNN model for trend, ARIMA(2, 0, 1) × $(2, 1, 1)_{12}$ model for seasonality, single-step forecast	8.71	11.94
DNN model for trend and seasonality, integral forecast	9.09	15.31
DNN model for trend, ARIMA(2, 0, 1) × $(2, 1, 1)_{12}$ model for seasonality, integral forecast	**6.81**	**10.90**
RNN model for trend, single-step forecast	17.02	23.72
RNN model for trend, integral forecast	13.94	16.66
RNN model for trend and seasonality, single-step forecast	8.37	14.67
RNN model for trend, ARIMA model for seasonality, single-step forecast	**6.72**	**10.53**
RNN model for trend and seasonality, integral forecast	10.51	16.17
RNN model for trend, модель ARIMA(2, 0, 1) × $(2, 1, 1)_{12}$ model for seasonality, single-step forecast	8.95	11.81
LSTM model for trend, single-step forecast	26.56	38.00
LSTM model for trend, integral forecast	23.35	31.07
LSTM model for trend and seasonality, single-step forecast	8.39	14.76
LSTM model for trend, ARIMA(2, 0, 1) × $(2, 1, 1)_{12}$ model for seasonality, single-step forecast	**6.87**	**10.58**
LSTM model for trend and seasonality, integral forecast	8.78	15.41
LSTM model for trend, ARIMA(2, 0, 1) × $(2, 1, 1)_{12}$ model for seasonality, integral forecast	7.30	11.13
GRU model for trend, single-step forecast	24.82	34.09
GRU model for trend, integral forecast	21.47	26.67
GRU model for trend and seasonality, single-step forecast	8.90	15.48
GRU model for trend, модель ARIMA(2, 0, 1) × $(2, 1, 1)_{12}$ model for seasonality, single-step forecast	7.88	11.24
GRU model for trend and seasonality, integral forecast	10.11	16.34
GRU model for trend, модель ARIMA(2, 0, 1) × $(2, 1, 1)_{12}$ model for seasonality, integral forecast	8.87	12.09

Based on an analysis of ACF/PACF[4] functions, a conclusion was drawn about the presence of seasonality in 12 months (which is confirmed by statistical tests) and ARIMA(p, d, q) mathematical models were selected. Their selection and analysis are detailed in [3, 30].

Note that the best forecasts are obtained for a combination of models (neural network model for trend, ARIMA for seasonality). Practically any model for the trend gives good results (the best results are obtained by the full-link network). The LSTM model performed better for single-step forecasts, the RNN and DNN models performed better for integral forecasts, while the GRU model worked well for both approaches. In this experiment, neural network models outperformed standard time series models.

Experiment 2 considers the index of real agricultural production in Russia for 2000–2020 (a detailed analysis of the series is presented in [30]). All the models considered were adapted for the 2000–2020 training period (the crisis years 2008 and 2014 were removed and the data agglomerated). The results of their forecasts for the test period (2021) are compared in Table 3.

[4] ACF—autocorrelation function; PCF—partial autocorrelation function.

According to ACF/PACF functions, it can be concluded that there is seasonality in 12 months (which is confirmed by statistical tests). Also, due to the presence of a spike in the PACF diagram, all ARIMA(p, d, q) models with orders p, q from 1 to 6 were tested. The ARIMA(2, 0, 1) model with annual seasonality of the form $(2, 1, 1)_{12}$ is used for a combination of models in which the standard ARIMA model is used to describe seasonality and the trend is specified using a neural network model or polynomial.

Note that the best forecasts are obtained for a combination of models (neural network model for trend, ARIMA for seasonality). At the same time, not all models give good results for the trend (RNN, DNN and LSTM give the best results). RNN and LSTM models performed better for single-step forecast, and DNN—for integral forecast. In this experiment, neural network models outperformed standard time series models.

Let us separately consider a series of exchange rate of exchange-traded shares: the shares of SberBank of the Russian Federation. It has heteroskedasticity: its mathematical expectation and dispersion change with time. This is confirmed by the McLeod–Li test (all components of the resultant vector are zero with an accuracy of 0.01) [31]. Due to the fact that it is non-seasonal, only two approaches are possible for each neural network system: to make a forecast for the entire test period at once (integral) or to make step-by-step forecasts, declaring each new step as part of the training sample, in order to move to the next point in time. They clearly indicate the absence of seasonality and the need to test second-order models. ARIMA(2, 1, 3) model was chosen for the series (the analysis of the series is given in [30]).

The forecasting results are presented in Table 4.

The best results are obtained by classical methods of series modeling: LOESS, ETS, and ARIMA models. Among neural network methods, the best result was shown by GRU model. In all cases, forecasts made one step ahead several times are better than a single forecast for a given period.

CONCLUSIONS

For seasonal time series, neural network-based models outperform standard models in terms of forecast accuracy for the test time period. The forecast accuracy of neural network models in all experiments was better than that of ARIMA/ETS models. The single-step forecast appears to be computationally less efficient than the integral forecast for the entire target period at once. However, it is not possible to specify precisely for which series the single-step or integral forecast is better in terms of quality.

Combined models, in which neural network models are used to model trend, and ARIMA model is used to model seasonality (when decomposed into trend, noise and seasonality, for example, using STL), almost always give a good result. More often than not, it is this model which provides the best result. At the same time, since the results are approximately equal, due to the lower complexity of construction and training, the RNN and full-link DNN models appear preferable.

When forecasting non-seasonal series, one-step forecasting is recommended (each predicted value is announced as part of the training sample to predict the next value). When forecasting the share price of SberBank of the Russian Federation, the best results were shown by the standard models and RNN.

Table 4. SberBank of the Russian Federation share price time series models

Time series model	MAE	RMSE
LOESS method	**0.004**	**0.006**
ARIMA (2, 1, 3)	**11.23**	**42.11**
ETS	**4.95**	**20.68**
DNN model for trend, single-step forecast	23.49	27.22
DNN model for trend, integral forecast	51.00	62.80
RNN model for trend, single-step forecast	16.42	21.69
RNN model for trend, integral forecast	80.53	86.39
LSTM model for trend, single-step forecast	49.74	59.32
LSTM model for trend, integral forecast	76.95	81.40
GRU model for trend, single-step forecast	**7.14**	**29.28**
GRU model for trend, integral forecast	24.66	85.05

When building neural networks modeling, the behavior of time series, several layers should be used (5–6 layers were used in this work). Networks with 1–2 layers do not extract features useful for forecasting even when the number of neurons in a layer is increased.

Authors' contribution. All authors equally contributed to the research work.

REFERENCES

1. Hyndman R.J., Athanasopoulos G. *Forecasting: principles and practice.* 3rd ed. OTexts; 2021. 442 p. ISBN-13 978-0987507136
2. Stock J.H., Watson M.W. *Introduction to Econometrics.* 3rd ed. Pearson; 2019. ISBN-13 978-9352863501
3. Kalugin T.R., Kim A.K., Petrusevich D.A. Analysis of the high order ADL(*p, q*) models used to describe connections between time series. *Russ. Technol. J*. 2020;8(2):7–22 (in Russ.). https://doi.org/10.32362/2500-316X-2020-8-2-7-22
4. Petrusevich D. Improvement of time series forecasting quality by means of multiple models prediction averaging. In: *Proceedings of the Third International Workshop on Modeling, Information Processing and Computing* (*MIP: Computing-2021*). 2021;2899:109–117. https://doi.org/10.47813/dnit-mip3/2021-2899-109-117
5. Beletskaya N., Petrusevich D. Linear combinations of time series models with minimal forecast variance. *J. Commun. Technol. Electron*. 2023;67(1):144–158. https://doi.org/10.1134/S1064226922130022
6. Box G., Jenkins G. *Time Series Analysis: Forecast and Management*. John Wiley & Sons; 2015. 712 p. ISBN 978-11185674918
7. Haykin S. *Neural Networks and Learning Machines*. Pearson Education; 2011. 936 p. ISBN 978-0133002553
8. Shi J., Jain M., Narasimhan G. *Time Series Forecasting* (*TSF*) *Using Various Deep Learning Models.* 2022. Available from URL: https://arxiv.org/abs/2204.11115v1, https://doi.org/10.48550/arXiv.2204.11115
9. Amalou I., Mouhni N., Abdali A. Multivariate time series prediction by RNN architectures for energy consumption forecasting. *Energy Rep*. 2022;8:1084–1091. https://doi.org/10.1016/j.egyr.2022.07.139
10. Aseeri A. Effective RNN-Based forecasting methodology design for improving short-term power load forecasts: application to large-scale power-grid time series. *J. Computational Sci*. 2023;68(4):101984. https://doi.org/10.1016/j.jocs.2023.101984
11. Ning Y., Kazemi H., Tahmasebi P. A comparative machine learning study for time series oil production forecasting: ARIMA, LSTM, and Prophet. *Comput. Geosci*. 2022;164(1):105126. https://doi.org/10.1016/j.cageo.2022.105126
12. Wang P., Zheng X., Ai G., Liu D., Zhu B. Time series prediction for the epidemic trends of COVID-19 using the improved LSTM deep learning method: Case studies in Russia, Peru and Iran. *Chaos, Solitons & Fractals*. 2020;140:110214. https://doi.org/10.1016/j.chaos.2020.110214
13. Arunkumar K.E., Kalaga D.V., Kumar M.S., Kawaji M., Brenza T.M. Comparative analysis of Gated Recurrent Units (GRU), long Short-Term memory (LSTM) cells, autoregressive Integrated moving average (ARIMA), seasonal autoregressive Integrated moving average (SARIMA) for forecasting COVID-19 trends. *Alexandria Eng. J*. 2022;61(10):7585–7603. https://doi.org/10.1016/j.aej.2022.01.011
14. Kumar B., Sunil, Yadav N. A novel hybrid model combining *βSARMA* and LSTM for time series forecasting. *Appl. Soft Comput*. 2023;134:110019. https://doi.org/10.1016/j.asoc.2023.110019
15. Abebe M., Noh Y., Kang Y.-J., Seo C., Kim D., Seo J. Ship trajectory planning for collision avoidance using hybrid ARIMA-LSTM models. *Ocean Eng*. 2022;256:111527. https://doi.org/10.1016/j.oceaneng.2022.111527
16. Cascone L., Sadiq S., Ullah S., Mirjalili S., Ur H., Siddiqui R., Umer M. Predicting household electric power consumption using multi-step time series with convolutional LSTM. *Big Data Res*. 2023;31:100360. https://doi.org/10.1016/j.bdr.2022.100360
17. Wang H., Zhang Y., Liang J., Liu L. DAFA-BiLSTM: Deep Autoregression Feature Augmented Bidirectional LSTM network for time series prediction. *Neural Netw*. 2023;157:240–256. https://doi.org/10.1016/j.neunet.2022.10.009
18. Zhao L., Mo C., Ma J., Chen Z., Yao C. LSTM-MFCN: A time series classifier based on multi-scale spatial–temporal features. *Computer Commun*. 2022;182(3):52–59. https://doi.org/10.1016/j.comcom.2021.10.036
19. Rasjid Z.E., Setiawan R., Effendi A. A Comparison: Prediction of Death and Infected COVID-19 Cases in Indonesia Using Time Series Smoothing and LSTM Neural Network. *Procedia Comput. Sci*. 2021;179(5):982–988. http://doi.org/10.1016/j.procs.2021.01.102
20. Dubey A.K., Kumar A., García-Díaz V., Sharma A.K., Kanhaiya K. Study and analysis of SARIMA and LSTM in forecasting time series data. *Sustain. Energy Technol. Assess*. 2021;47:101474. https://doi.org/10.1016/j.seta.2021.101474
21. Wu Z., Yin H., He H., Li Y. Dynamic-LSTM hybrid models to improve seasonal drought predictions over China. *J. Hydrol*. 2022;615:128706. https://doi.org/10.1016/j.jhydrol.2022.128706
22. Yan Y., Wang X., Ren F., Shao Z., Tian C. Wind speed prediction using a hybrid model of EEMD and LSTM considering seasonal features. *Energy Rep*. 2022;8:8965–8980. https://doi.org/10.1016/j.egyr.2022.07.007
23. Bian S., Wang Z., Song W., Zhou X. Feature extraction and classification of time-varying power load characteristics based on PCANet and CNN+Bi-LSTM algorithms. *Electric Power Systems Research*. 2023;217(6):109149. https://doi.org/10.1016/j.epsr.2023.109149

24. Sangiorgio M., Dercole F. Robustness of LSTM neural networks for multi-step forecasting of chaotic time series. *Chaos, Solitons & Fractals*. 2020;139(8):10045. https://doi.org/10.1016/j.chaos.2020.110045
25. Liu X., Lin Z., Feng Z. Short-term offshore wind speed forecast by seasonal ARIMA – A comparison against GRU and LSTM. *Energy*. 2021;227:120492. http://doi.org/10.1016/j.energy.2021.120492
26. Shahid F., Zameer A., Muneeb M. Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM. *Chaos, Solitons & Fractals*. 2020;140:110212. https://doi.org/10.1016/j.chaos.2020.110212
27. Wang J., Wang P., Tian H., Tansey K., Liu J., Quan W. A deep learning framework combining CNN and GRU for improving wheat yield estimates using time series remotely sensed multi-variables. *Comput. Electron. Agric*. 2023;206(4):107705. https://doi.org/10.1016/j.compag.2023.107705
28. Hua H., Liu M., Li Y., Deng S., Wang Q. An ensemble framework for short-term load forecasting based on parallel CNN and GRU with improved ResNet. *Electric Power Syst. Res*. 2023;216(3):109057. https://doi.org/10.1016/j.epsr.2022.109057
29. Zhang D., Sun W., Dai Y., Liu K., Li W., Wang C. A hierarchical early kick detection method using a cascaded GRU network. *Geoenergy Sci. Eng*. 2023;222(3):211390. https://doi.org/10.1016/j.geoen.2022.211390
30. Gramovich I.V., Musatov D.Yu., Petrusevich D.A. Implementation of bagging in time series forecasting. *Russ. Technol. J*. 2024;12(1):101–110 (in Russ.). https://doi.org/10.32362/2500-316X-2024-12-1-101-110
31. McLeod A., Li W. Diagnostic checking ARMA time series models using squared residual autocorrelations. *J. Time Ser. Anal*. 1983;4(4):269–273. https://doi.org/10.1111/j.1467-9892.1983.tb00373.x

About the authors

Bakhtierzhon Pashshoev, Student, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: bahtiyorposhshoev@gmail.com. https://orcid.org/0009-0000-2019-2642

Denis A. Petrusevich, Cand. Sci. (Phys.-Math.), Associate Professor, Higher Mathematics Department, Institute of Artificial Intelligence, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: petrusevich@mirea.ru, petrdenis@mail.ru. Scopus Author ID 55900513600, ResearcherID AAA-6661-2020, RSCI SPIN-code 7999-6345, https://orcid.org/0000-0001-5325-6198

Об авторах

Пашшоев Бахтиёржон, студент, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: bahtiyorposhshoev@gmail.com. https://orcid.org/0009-0000-2019-2642

Петрусевич Денис Андреевич, к.ф.-м.н., доцент, кафедра высшей математики, Институт искусственного интеллекта, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: petrusevich@mirea.ru, petrdenis@mail.ru. Scopus Author ID 55900513600, ResearcherID AAA-6661-2020, SPIN-код РИНЦ 7999-6345, https://orcid.org/0000-0001-5325-6198

Translated from Russian into English by Lyudmila O. Bychkova
Edited for English language and spelling by Dr. David Mossop

MIREA – Russian Technological University.
78, Vernadskogo pr., Moscow, 119454 Russian Federation.
Publication date July 30, 2024.
Not for sale.

МИРЭА – Российский технологический университет.
119454, РФ, г. Москва, пр-т Вернадского, д. 78.
Дата опубликования 30.07.2024 г.
Не для продажи.