ORIGINAL PAPER

Comparative Analysis of ARIMA and LSTM Predictive Models: Evidence from Russian Stocks

A.V. Alzheev[a], **R.A. Kochkarov**[b]
Financial University, Moscow, Russia
[a] https://orcid.org/0000-0001-8944-5679;
[b] https://orcid.org/0000-0003-3186-3901

ABSTRACT

The article **aims** to find the best time series predictive model, considering the minimization of errors and high accuracy of the prediction. The authors performed the comparative analysis of the most popular "traditional" econometric model ARIMA and the deep learning model LSTM (Long short-term memory) based on a recurrent neural network. The study provides a mathematical description of these predictive models. The authors developed algorithms for predicting time series based on the "Rolling forecasting origin" approach. These are Python-based algorithms using the Keras, Theano and Statsmodels libraries. Stock quotes of Russian companies Alrosa, Gazprom, KamAZ, NLMK, Kiwi, Rosneft, VTB and Yandex for the period from June 2, 2014 to November 11, 2019, broken down by week, served as input data. The research **results** confirm the superiority of the LSTM model, where the RMSE error is 65% less than with the ARIMA model. Therefore, an LSTM model-based algorithm is more preferable for the better quality of time series prediction.

Keywords: ARIMA; LSTM; predictive models; stocks; analysis; stock quote prediction; algorithms

For citation: **Alzheev A.V., Kochkarov R.A. Comparative analysis of ARIMA and LSTM predictive models: Evidence from Russian stocks. *Finance: Theory and Practice*. 2020;24(1):14-23. (In Russ.). DOI: 10.26794/2587-5671-2020-24-1-14-23**

INTRODUCTION

Time series prediction is difficult, mostly due to unexpected changes in the economy and information asymmetry. Due to market volatility, the quality of regression predictive modeling has recently been a major concern. Therefore, it must be carefully evaluated.

A time series is a sequence of observation values taken at successive equally spaced points in time. Examples of a time series may include production output, number of insurance products sold, floods, etc. Time series are widely used in statistics, signal processing, pattern recognition, econometrics, finance, astronomy, engineering, and most other human activities, which involve temporal measurements. Time series analysis comprises methods for analyzing time series data in order to extract meaningful information to predict future values based on previously observed values.

Time series data have a natural temporal ordering. This makes time series analysis distinct from cross-sectional studies (e.g. explaining people's wages by reference to their gender). Time series analysis is also distinct from spatial data analysis (the dependence of wages on the region). Time series models reflect the fact that observations close together in time will be more closely related than observations further apart.

The aim of this work is the best time series predictive model, considering the minimization of errors and high accuracy of the prediction.

In the context of predicting financial time series, the ARIMA model (autoregressive integrated moving average model, sometimes called the Box-Jenkins model) is one of the most popular and frequently used time series models [1]. Popular models of deep learning are RNN (recurrent neural network) [2] and its modification LSTM (long short-term memory), introduced by Z. Hochreiter and J. Schmidhuber in 1997 [3]. Deep learning allows you to find complex patterns in sequential spatial chains. For example, a recurrent neural network is used to recognize handwritten text and speech. The LSTM model, in particular, was used to predict the volatility of the S&P 500 index [4–6].

This article compares the ARIMA and LSTM models, and the criterion for comparison is minimizing the prediction errors. The ARIMA model was selected due to its ability to work with non-stationary data. Among other deep learning models, LSTM was chosen as the most suitable model for predicting time series with the capacity to preserve memory for a long period of time.

ALGORITHMIC TRADING

Over the past few decades, algorithmic trading has been actively developing due to a combination of factors: rapid development of machine learning methods, development of data processing and analysis technologies, growing capacities for more data storage and processing. Besides, the complexity of trading system algorithms used by market participants is growing, as they compete not only with those who do not use automated systems, but also with each other. Therefore, the study of applying various machine learning algorithms to algorithmic trading problems is an urgent issue. These studies are of interest not only to algorithmic trading companies, such as hedge funds, but also to the scientific community: applying machine learning algorithms to the field in question can bring new knowledge to the development of machine learning as a branch of computer science that can be applied to other subject areas.

The theoretical value of the work is the development of research on applying machine learning methods to algorithmic trading. Moreover, similar methods can be applied to other subject areas when searching for a strategy for the most accurate prediction of the next value of the time series to maximize profits. The practical value of the work is the possibility to create software products for stock trading based on the studied algorithms.

MACHINE LEARNING IN THE STOCK MARKET

Stock market is a public trading platform for buying and selling securities of various companies. The successful prediction of a stock's future price is an important economic task for a wide range of companies, since could yield significant profit. From the very beginning of stock markets, people have been looking for and developing ways to predict stock quotes as accurately as possible. As a result, there are three main approaches: fundamental analysis, technical analysis and technological methods. Sometimes, these approaches have mutually exclusive results within one task (some predict the future value; the others predict the direction of movement for a given period). However, all of them are used for trading on stock markets. Improving prediction accuracy remains relevant for all companies in stock markets. This paper is devoted to applying machine learning (including deep learning) to prediction in the stock market. Both classical machine learning and neural networks are employed as predictive tools. However, reinforced learning and genetic algorithms have not become so popular and developed as many classical algorithms (random forest, support vector method). A feedforward neural network and a backpropagation neural network (as a learning method) are one of the known types of neural networks. In trading on the stock market, neural networks can imitate the actions of an agent performing certain tasks. Data preprocessing is an important step in building neural networks. Many researchers propose their own solutions, since there are no standard procedures for preprocessing input data due to the wide variety of both data and types of preprocessing. On the contrary, output data within a specific task can be uniquely determined and take the form of financial or economic indicators. As for the evaluation of the results, comparing the results of the algorithm with the values from the test sample is most common.

MACHINE LEARNING METHODS

The most traditional way to apply machine learning to algorithmic trading is to learn with a teacher (supervised learning). The target variable, whose value is predicted by the algorithm, is the change in the price of an asset over a certain period of time or, alternatively, market volatility. At the beginning, machine learning prediction on stock exchanges often used simple and well-studied decision trees of the support vector machine. Technical market indicators are usually the input features of such algorithms. After neural networks were successfully applied to other tasks, such as natural language processing, they were used in algorithmic trading, for example, to predict price changes in the market affected by emerging political or economic news. It became possible to extract market signals from unstructured information. Reinforcement learning is one of the fast-growing method of machine learning. It helped to achieve such results as the ability of a computer to play various games on the Atari simulator without knowing the game rules at a level exceeding the level of a person.

Reinforcement learning had been applied to the tasks of algorithmic trading before. However, supervised learning was more common, since it helped achieve better results. Recently, the success achieved through reinforcement learning in other areas has increased interest in applying this type of algorithms to algorithmic trading.

Genetic algorithms are successfully used in a wide range of application areas, for example, robotics. One can use genetic algorithms in generalized problems, such as building ensembles of machine learning algorithms to improve the accuracy of individual algorithms and building decision trees. Genetic algorithms also show their effectiveness in such tasks. In algorithmic trading, they were initially used mainly to generate the so-called trading rules. This method employs genetic algorithms to create functions of technical market indicators, the most accurate in predicting price changes of an asset over a certain period. Currently, genetic algorithms are used to build automated trading systems.

TIME-SERIES RESEARCH

Time-series analysis is an interesting field of research applied in numerous practical fields such as business, economics, finance and computer science. Time-series analysis aims to study its dynamics, to build a model describing data structure and, finally, to predict future values of the series. Building an effective model with the highest possible accuracy is critical.

Econometrics has traditionally been involved in time series prediction by various methods. The ARIMA model has long been considered the standard in this field. However, the common ARIMA model has some constraints. For example, it is difficult to model nonlinear relationships between variables by means of the ARIMA model. The model presupposes a constant standard deviation of errors, not observed in practice. An integrated approach with the GARCH model (Generalized auto-regressive conditional heteroscedasticity) helps ease this restriction, but building and optimizing the model becomes more difficult. In this sense, the GARCH model is intended to clarify volatility clustering in financial markets [7].

Traditional machine learning has suggested new approaches to prediction. Algorithms such as the Support Vector Machines (SVM) and the Random Forest (RF) are well-deserved attention of experts in many areas, including finance. Application of deep learning algorithms was the next step.

To predict time-series, deep learning most frequently uses the LSTM model, a special type of convolutional neural network. Despite the relative novelty of this method, the deep learning approach has gained widespread popularity among researchers. For example, K. Krauss and his colleagues used predictive approaches — deep learning, gradient-boosted-trees and random forest — to model the S&P 500 [8]. The results are the conclusions about the low productivity of deep learning models and their difficulty. Various models were compared based on economic data [9, 10].

ARIMA AND LSTM MODELS AND PREDICTIVE CALCULATIONS

A mathematical description of the ARIMA and LSTM predictive models is following.

ARIMA is a generalized ARMA (Autoregressive moving average) model that combines AR (Autoregressive) and MA (Moving average) processes. As indicated in the acronym of the model, ARIMA (*p, d, q*) consists of the following parts [1, 11]:

AR — autoregressive. A regression model that uses the relationship between observations and the number of integrated observations (*p*);

I — integrated. To ensure stationarity by taking differences (*d*);

MA — moving average. An approach that analyzes the relationship between observations and residuals when applying the model to integrated observations (*q*).

A simple form of an AR model of order p, i.e., AR(*p*), can be written as a linear process given by:

$$x_t = c + \sum_{i=1}^{p} \varphi_i x_{t-1} + \in_i, \tag{1}$$

where x_t — represents the stationary variable; c is the constant; φ_i — are autocorrelation coefficients; and $\in_i$ — the residuals, are the Gaussian white noise series with mean zero.

A simple form of an MA model of order *q*, i.e., MA(*q*), can be written as follows:

$$x_t = \mu + \sum_{i=0}^{q} \theta_i \in_{t-1}, \tag{2}$$

where μ — is the mathematical expectation of the process (usually assumed to be equal to zero); θ_i — is weights; θ_0 — assumed to be equal to 1; $\in_t$ — is the Gaussian white noise with mean zero. The combination of these two models provides an ARIMA model of order (*p, q*):

$$x_t = c + \sum_{i=1}^{p} \varphi_i x_{t-1} + \in_i + \sum_{i=0}^{q} \theta_i \in_{t-1}. \tag{3}$$

The parameters *p* and *q* are called the order of AR and MA, respectively. ARIMA allows predictions on non-stationary data due to the integration introduced into the model. This is achieved by taking differences — subtracting the levels of the time series from each other.

Given the seasonality of time series, short-term components are likely to make a significant contribution to the model. Thus, the model should also consider seasonality — seasonal ARIMA (seasonal ARIMA — SARIMA). The most important steps are to evaluate the coefficients of the model. If the variance grows over time, it is necessary to use stabilizing variance transformations and taking differences. Using the autocorrelation function and the private autocorrelation function, one should measure the linear relationship between the observations and *q*.

LSTM is a type of the convolutional neural network able to “remember” the values of the previous observations for future use. For more information about the LSTM model, we will provide some definitions and explanations [2, 3].

Artificial neural network is a neural network consisting of at least three layers: input, hidden and output. The number of variables in the dataset determines the dimension or number of nodes in the input layer. These nodes are connected by edges. Each edge carries some weight, on which it depends whether the signal can pass through the layer or not. When an artificial neural network learns, the weights based on the data are changing. In hidden layers, nodes use an activation function (for example, a sigmoid) on a weighted sum of input data to convert it to the output (in this case, predictions). The output layer generates a probability vector for various predicted values and selects one with the least error. Weights cannot be optimal immediately, so the neural network learns considering the result of the previous iteration. These iterations are called epochs, and the weights change until an optimal (given) value is reached.

Recurrent neural network (convolutional neural network) is a special neural network, aiming to predict the next observation in the series. The idea behind the RNN is the desire to extract useful information from a series of observations to make predictions. Therefore, it is necessary to remember earlier observations. In the RNN model, the inner layer serves to store information from previous observations of the series. The main problem is to remember a small number of previous observations, which is not suitable for long (financial, economic) periods. LSTM networks were developed to solve this problem.

Long short-term memory neural network is an artificial convolutional neural network used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It cannot only process single data points (such as images), but also entire sequences of data (such as speech or video). Therefore, LSTM is applicable to tasks such as handwriting recognition, speech recognition and anomaly detection in large data streams (network traffic, banking transactions). LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. This time lag leads to exploding and vanishing gradient problems that can be encountered when training traditional RNNs. Insensitivity to gap length is an advantage of LSTM over RNNs.

There are three layers in LSTM networks:

1. Forget gate — a number from 0 to 1 is output, where 1 indicates the need for complete storage, and 0 completely erases from memory.
2. Memory gate selects which data to store. First, a sigmoid layer helps select values to store.
3. Output gate selects information from each cell that has stored it.

COMPARATIVE ANALYSIS OF ARIMA AND LSTM MODELS

To compare the ARIMA and LSTM models, the authors conducted experiments on financial data in the form of time series. The main research question is which algorithm, ARIMA or LSTM, allows better time series prediction.

The authors collected[1] historical data on stock prices of Russian enterprises: Alrosa, Gazprom, KAMAZ, NLMK, QIWI, Rosneft, VTB and Yandex. Each data set contains values such as ‘Open’, ‘High’, ‘Low’, ‘Close’, ‘Volume’. For analysis, we selected the ‘Close’ value for the period from June 2, 2014 to November 11, 2019, broken down by week. We also divided the data into training and test sets, in the ratio of 70% to 30%, not mixed. We used the training dataset to train the model, and the test set to assess the quality of its training. *Fig. 1* presents the graphs of the collected data.

The RMSE (Root Mean Square Error) is used to assess the quality of model prediction. It measures the difference between the true and predicted values. The formula of the indicator is as follows [12]:

$$RMSE = \sqrt{\frac{1}{N}\sum_{1}^{N}(x_i - \widehat{x}_i)^2}\,, \tag{4}$$

where N — is the total number of observations; and $\sum_{1}^{N}(x_i - \widehat{x}_i)$ represents the square of the sum of the differences of the true value and the prediction of the model. The main advantage is that the RMSE penalizes the largest errors.

DEVELOPMENT OF ARIMA AND LSTM ALGORITHMS

To predict time series, the authors developed the algorithms based on the “Rolling forecasting origin” [13]. The algorithm is based on pre-

[1] URL: https://www.finam.ru/profile/moex-akcii/gazprom/export/ (accessed on 14.11.2019).

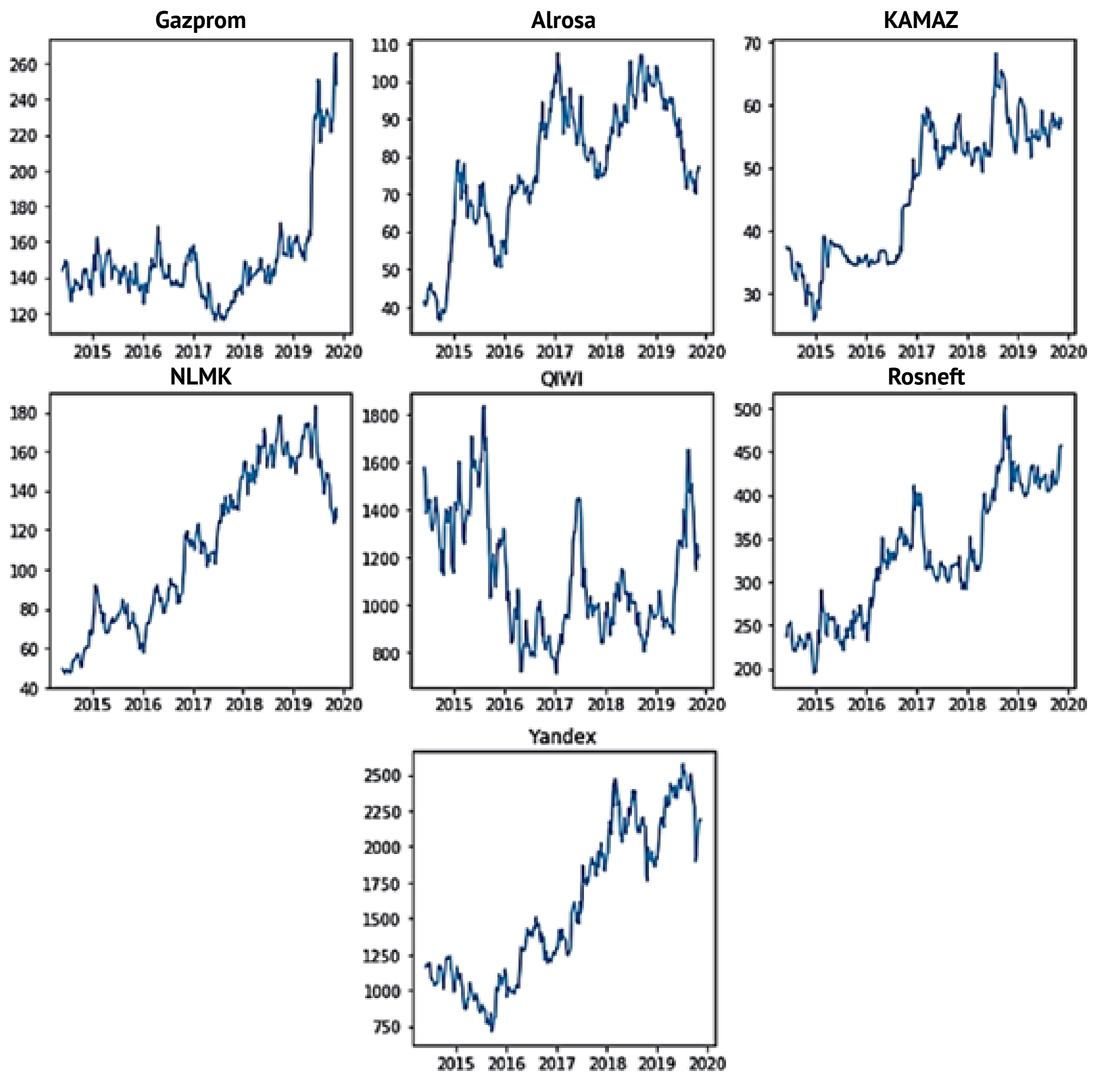


Fig. 1. **Stock performance of Russian companies**

Source: compiled by the authors.

dicting the next point in the time series of each data set. The approach uses training data sets, each containing one observation more than the previous one. There are several different variations of this algorithm, among which are:

1. One-step prediction without revaluation. The model evaluates a training set and then calculates the prediction one step ahead.
2. Multi-step prediction without revaluation. Similar to the previous one, however, the prediction is given for the next few steps.
3. Multi-step prediction with revaluation. The prediction is calculated several steps forward, however, the model is trained anew at each iteration before the prediction.

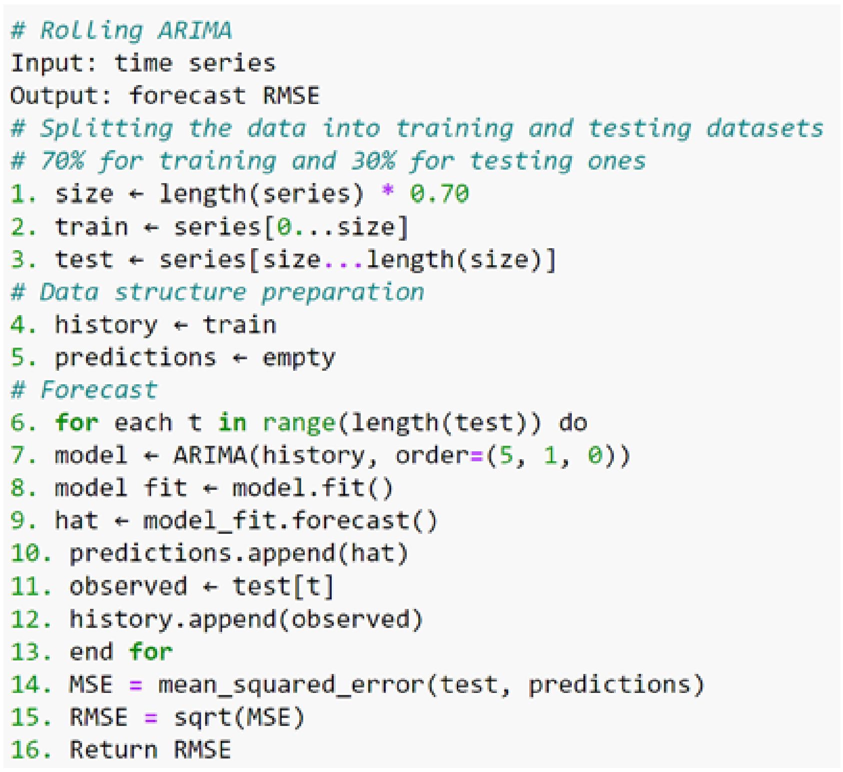
This article uses an algorithm building a new model every time new data comes in. The algorithms are implemented in Python[2] with the connected libraries Keras, Theano and Statsmodels.

Fig. 2 shows a sliding window algorithm for the ARIMA model. The ARIMA algorithm is divided into the following stages:

1) the input data (each financial time series) is divided into training and test sets;

2) additional structures are created (arrays where records are stored and changes are made

[2] URL: https://www.python.org/ (accessed on 15.01.2020).



```
# Rolling ARIMA
Input: time series
Output: forecast RMSE
# Splitting the data into training and testing datasets
# 70% for training and 30% for testing ones
1. size ← length(series) * 0.70
2. train ← series[0...size]
3. test ← series[size...length(size)]
# Data structure preparation
4. history ← train
5. predictions ← empty
# Forecast
6. for each t in range(length(test)) do
7. model ← ARIMA(history, order=(5, 1, 0))
8. model fit ← model.fit()
9. hat ← model_fit.forecast()
10. predictions.append(hat)
11. observed ← test[t]
12. history.append(observed)
13. end for
14. MSE = mean_squared_error(test, predictions)
15. RMSE = sqrt(MSE)
16. Return RMSE
```

Fig. 2. **Sliding window algorithm for ARIMA model**

Source: compiled by the authors.

about "current" historical data, predictions) to train the model;

3) the model is trained on the current data set and makes a prediction, which is then written to the predictions array and changes the history array;

4) the RMSE metric is calculated for the ARIMA model.

Fig. 3 shows a sliding window algorithm for the LSTM model.

Unlike regression models, there is a sequence of dependencies between input variables in the time data analysis. Convolutional neural networks capture such dependencies very well.

The LSTM algorithm is divided into the following steps:

1) the data is divided into training and test sets;

2) random_state parameter is fixed to reproduce the results;

3) fit_lstm function is defined, the data set, the number of epochs and the number of neurons are taken for arguments;

4) the hidden layer is created;

5) the optimization algorithm and loss function (Adam and MSE) are specified;

6) then, the model is trained, while its internal state is restored (code, line 12) to the original one each time;

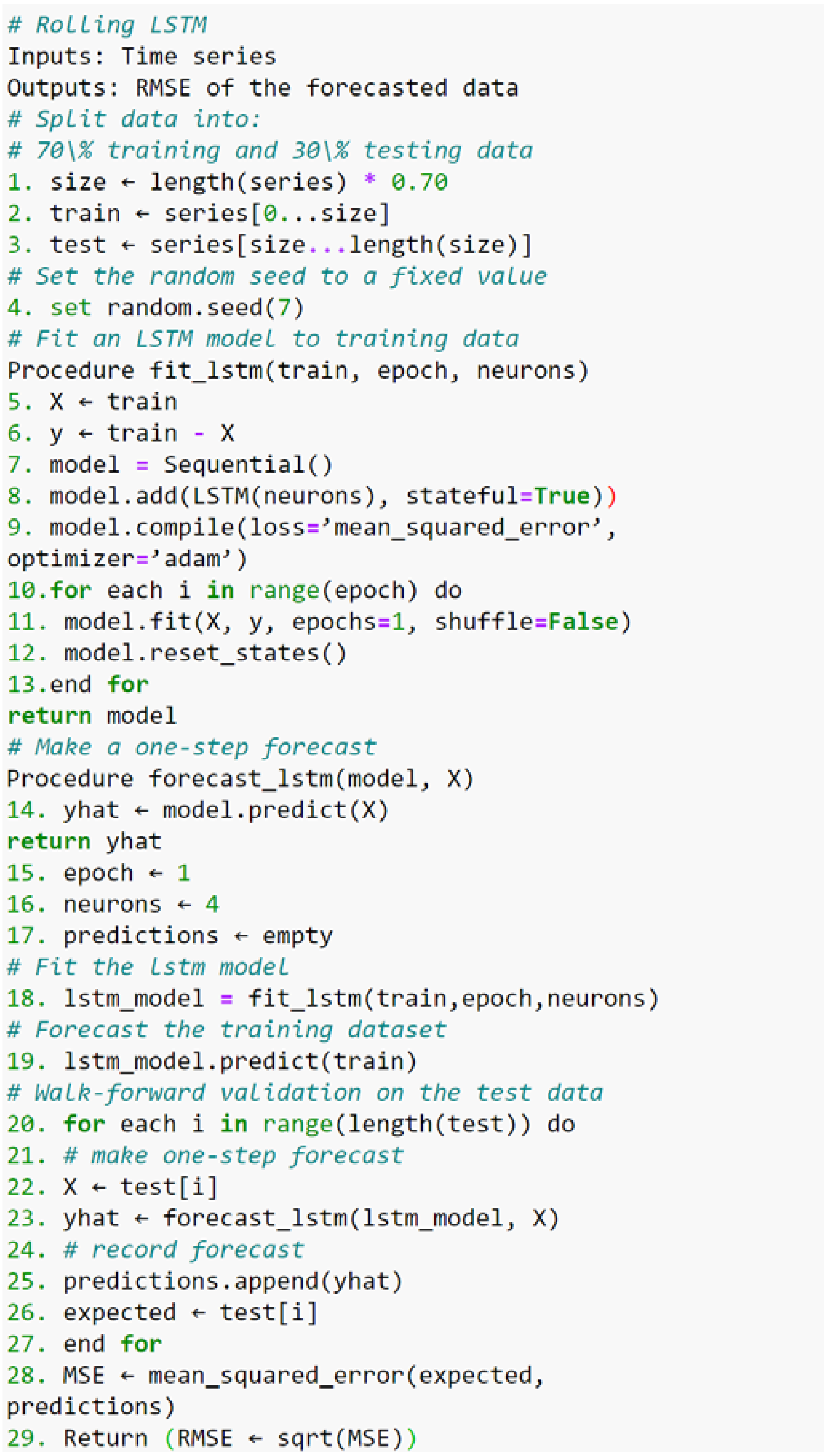
7) a prediction is made and the RMSE metric is calculated for the LSTM model.

ALGORITHM PERFORMANCE RESULTS

Fig. 4 presents algorithm performance results. On the financial time series, the LSTM model performed better than the ARIMA model. The values of the RMSE models are 10.8 and 26.7, respectively. Using the LSTM model reduces the RMSE value by 65% compared to the ARIMA model.

CONCLUSIONS

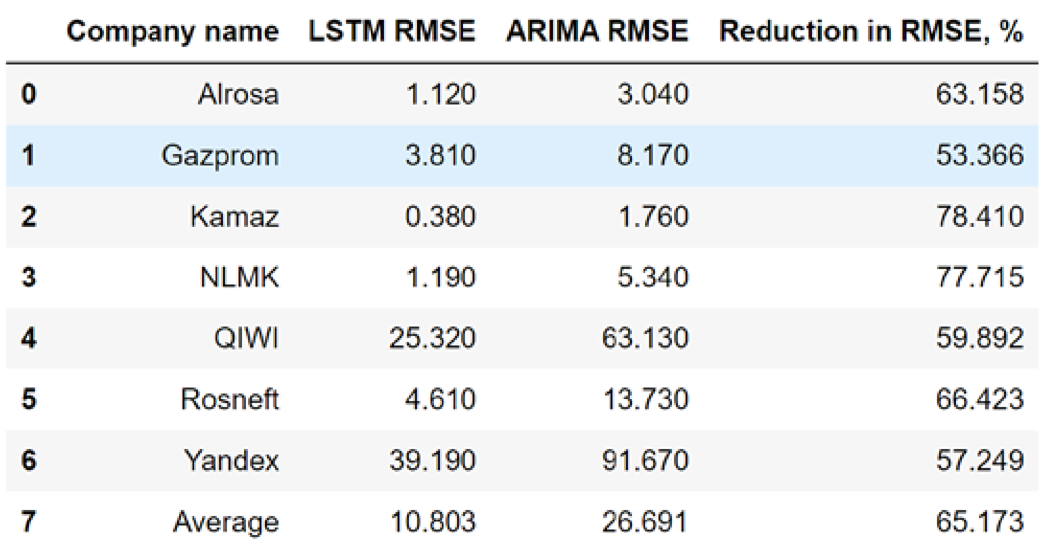
The comparative analysis of the two algorithms based on the LSTM and ARIMA models determined the superiority of the LSTM model over the ARIMA model in terms of minimizing the RMSE standard error. The RMSE error value of



```
# Rolling LSTM
Inputs: Time series
Outputs: RMSE of the forecasted data
# Split data into:
# 70\% training and 30\% testing data
1. size ← length(series) * 0.70
2. train ← series[0...size]
3. test ← series[size...length(size)]
# Set the random seed to a fixed value
4. set random.seed(7)
# Fit an LSTM model to training data
Procedure fit_lstm(train, epoch, neurons)
5. X ← train
6. y ← train - X
7. model = Sequential()
8. model.add(LSTM(neurons), stateful=True))
9. model.compile(loss='mean_squared_error',
optimizer='adam')
10.for each i in range(epoch) do
11. model.fit(X, y, epochs=1, shuffle=False)
12. model.reset_states()
13.end for
return model
# Make a one-step forecast
Procedure forecast_lstm(model, X)
14. yhat ← model.predict(X)
return yhat
15. epoch ← 1
16. neurons ← 4
17. predictions ← empty
# Fit the lstm model
18. lstm_model = fit_lstm(train,epoch,neurons)
# Forecast the training dataset
19. lstm_model.predict(train)
# Walk-forward validation on the test data
20. for each i in range(length(test)) do
21. # make one-step forecast
22. X ← test[i]
23. yhat ← forecast_lstm(lstm_model, X)
24. # record forecast
25. predictions.append(yhat)
26. expected ← test[i]
27. end for
28. MSE ← mean_squared_error(expected,
predictions)
29. Return (RMSE ← sqrt(MSE))
```

Fig. 3. **Sliding window algorithm for LSTM model**

Source: compiled by the authors.



	Company name	LSTM RMSE	ARIMA RMSE	Reduction in RMSE, %
0	Alrosa	1.120	3.040	63.158
1	Gazprom	3.810	8.170	53.366
2	Kamaz	0.380	1.760	78.410
3	NLMK	1.190	5.340	77.715
4	QIWI	25.320	63.130	59.892
5	Rosneft	4.610	13.730	66.423
6	Yandex	39.190	91.670	57.249
7	Average	10.803	26.691	65.173

Fig. 4. **ARIMA and LSTM algorithm execution results**

Source: compiled by the authors.

the LSTM model is less than the corresponding error for the ARIMA model by an average of 65%.

Currently developing big data analysis technologies, machine learning algorithms and, in particular, deep learning algorithms are gaining popularity among researchers in various fields of science. The main issue is the accuracy and adequacy of new approaches compared to traditional methods. The article compares the prediction accuracy of the ARIMA and LSTM models, representing two different methods. Both models were applied on various sets of financial data — the time series of closing prices at the end of the week. The results show the superiority of the LSTM model.

There are numerous tasks of quality assessment of models. The authors mentioned only some aspects. Studying and applying new algorithms for processing big data based on neural networks, large graphs and machine learning seems appropriate to study the financial market instruments of Russian companies. In this sense, an aggregated approach combining different approaches and algorithms seems promising.

ACKNOWLEDGEMENTS

The article was supported by the RFBR grant 18–00–01103. Financial University, Moscow, Russia.

REFERENCES

1. Magnus Ya.R., Katyshev P.K., Peresetsky A.A. Econometrics. Beginner course. Moscow: Delo; 2007. 504 p. (In Russ.).
2. Schmidhuber J. Habilitation. System modeling and optimization. Postdoctoral thesis. Munich: Technical University of Munich; 1993. 209 p. (In German).
3. Hochreiter S., Schmidhuber J. Long short-term memory. *Neural Computation*. 1997;9(8):1735–1780. DOI: 10.1162/neco.1997.9.8.1735
4. Brownlee J. Time series prediction with LSTM recurrent neural networks in Python with Keras. Machine Learning Mastery. 2016. URL: https://machinelearning mastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras
5. Gers F.A., Schmidhuber J., Cummins F. Learning to forget: Continual prediction with LSTM. *Neural Computation*. 2000;12(10):2451–2471. DOI: 10.1162/089976600300015015
6. Kratovich P.V. Neural networks and ARIMA models for predicting quotes. *Programmnye produkty i sistemy = Software & Systems*. 2011;(1):95–98. (In Russ.).

7. Garcia F., Guijarro F., Moya I., Oliver J. Estimating returns and conditional volatility: A comparison between the ARMA-GARCH-M models and the backpropagation neural network. *International Journal of Complex Systems in Science*. 2012;1(2):21–26.
8. Krauss C., Do X.A., Huck N. Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research*. 2017;259(2):689–702. DOI: 10.1016/j.ejor. 2016.10.031
9. Chung J., Lee D., Seo Y., Yoo C.D. Deep attribute networks. NIPS Workshop on Deep Learning and Unsupervised Feature Learning. 2012;3. URL: http://www.eng.uwaterloo.ca/~jbergstr/files/nips_dl_2012/Paper%2011.pdf
10. Fischer T., Krauss C. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*. 2018;270(2):654–669. DOI: 10.1016/j.ejor.2017.11.054
11. Eliseeva I.I., ed. Econometrics. Moscow: Finansy i statistika; 2006. 576 p. (In Russ.).
12. Armstrong J.S., Collopy F. Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting*. 1992;8(1):69–80. DOI: 10.1016/0169–2070(92)90008-W
13. Hyndman R.J., Athanasopoulos G. Forecasting: Principles and practice. 2nd ed. Melbourne: OTexts; 2018. 382 p.

ABOUT THE AUTHORS

Andrei V. Alzheev — Master's student, Department of Data Analysis, Decision Making and Financial Technology, Financial University, Moscow, Russia
alzheev@gmail.com

Rasul A. Kochkarov — Cand. Sci. (Econ.), Assoc. Prof., Department of Data Analysis, Decision Making and Financial Technology, Financial University, Moscow, Russia
rasul_kochkarov@mail.ru

The article was submitted on 18.11.2019; revised on 17.12.2019 and accepted for publication on 20.12.2019.
The authors read and approved the final version of the manuscript.