Stock Prices Dynamics Forecasting with Recurrent Neural Networks

1st Tatyana Vasyaeva
Automated Control Systems
Donetsk National Technical University
Donetsk, Ukraine
vasyaeva@gmail.com

2nd Tatyana Martynenko
Automated Control Systems
Donetsk National Technical University
Donetsk, Ukraine
tatyana.v.martynenko@gmail.com

3rd Sergii Khmilovyi
Automated Control Systems
Donetsk National Technical University
Donetsk, Ukraine
hmelevoy.sergey@gmail.com

4th Natalia Andrievskaya
Automated Control Systems
Donetsk National Technical University
Donetsk, Ukraine
nataandr@yandex.ru

***Abstract*—The application of deep neural networks was examined in the area of stock prices forecasting of pharmacies chain "36 and 6". The learning sample formation in the time series area was shown and the neural network architecture was proposed. The neural network for exchange trade forecasting using Python's Keras Library was developed and trained. The basic parameters setting of algorithm have been carried out.**

***Keywords*—Machine Learning, Deep Learning, Recurrent Neural Networks, Simple Recurrent Neural Network, Update Gate and Reset Gate, Long short-term memory, Time Series, Stock Prices.**

I. Introduction

Effective actions on the stock exchange are connected with a careful analysis of all that's happening on the market. Good and effective prediction systems for stock market help traders, investors, and analyst by providing supportive information like the future direction of the stock market. In order to make the forecasting as reliable as possible, and the forecasts – well grounded, traders use exchange analysis. Widely used methods of analysis, to which most market participants are accustomed [1], are not always effective. That's why over recent years, financial analysts have become of great interest in such a direction as machine learning [2] and in particular artificial neural networks [3] . In contrast to the classical methods of exchange analysis [1], which involve the implementation of ready-made algorithms, machine learning allows the system to learn how to recognize patterns and make forecasts.

In our work we will consider the problem of predicting the future stock prices of the pharmacies chain "36 and 6" based on the values of their stock prices in the past, as well as additional data of exchange trade for the period under report. Future stock price prediction is probably the best example of time series forecasting [4], [5].

II. Data Representation

The data that we are going to use for this article can be downloaded from servers "FINAM" and their web resource [6]. From the list of securities (instruments) provided by "FINAM" company we will take the data of quotations of the pharmacies chain "36 and 6". The received information provided by Moscow Exchange PJSC has the following format (Table I): date, opening price, maximum price, minimum price, closing price, volume. Usually datasets for working with time series

Table I
Data representation.

OPEN	HIGH	LOW	CLOSE	VOL
13.4100	13.7500	13.4000	13.5900	255490
13.4800	13.6900	13.3300	13.4200	151960
13.5400	14.6100	13.4400	14.5200	433440
14.6900	14.9100	14.4100	14.7600	178030
...	...	...	...	...
14.7500	15.2300	14.6500	15.1000	151950
14.8800	14.9500	14.6500	14.7900	77570

form a "sliding window" [4], [5] with a width equal to the depth of immersion. After the data are prepared in this way they will have such form (see Table II). As a rule of thumb, whenever you use a neural net-work, you should normalize or scale your data. We normalize the obtained data and divide it into parts. A common dataset

Table II
Training data.

№	OPEN	HIGH	LOW	CLOSE	VOL
1					
2					
...	...	...	...	...	...

after the transformations shown in Tab. II, with depth of immersion =100 has a size of 5x100x1260. To build the model, we will use a set consisting of 851 simples (of 1260 total), and under the validation set we will allocate 1% of this data. As a result, we get (see Tab. III).

Table III
TRAINING, VALIDATION AND TESTING DATASET.

Training	Validation	Testing
...	...	...
765 Data Points	86 Data Points	309 Data Points

III. MATERIALS AND METHODS

Nowadays deep neural networks become one of the most popular approaches to solving a wide variety of problems [7]–[10]. There is no unambiguous definition of what a deep neural network is. In this work, the term deep neural network will be understood as a neural network that contains more than one hidden layer. To train neural networks, including deep ones, we use the error back-propagation algorithm [3] based on the gradient descent method.

Recurrent Neural Networks (RNN). RNN [11]– are the networks with loops in them, allowing information to persist. There is a chunk of neural network (See Fig. 1). The network takes the input value xi and returns value hi. A loop allows information to be passed from one step of the network to the next. A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a subsequent copy. If we unroll the loop, we will get the following (see Fig. 1). So, RNN is a deep neural network. Simple RNN (see Fig. 2) is the

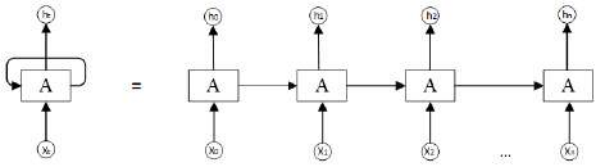

Figure 1. The fragment of recurrent neural network and an unrolled recurrent neural network..

RNN, where output is calculate as simple multiplication of Input (xi) and Previous Output (hi-1). Passed through Tanh activation function. No Gates present. Deep neural

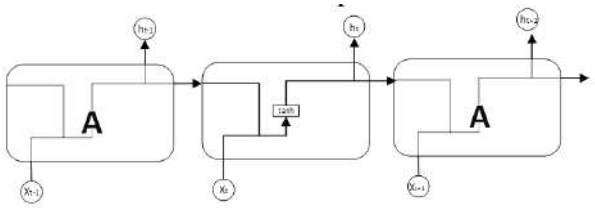

Figure 2. The repeating module in Simple RNN.

networks with many hidden layers are difficult to train because of the vanishing gradient problem. The problem of vanishing gradient can be solved by the architecture of a recurrent neural network called a network of long short–term memory [11], [12] and Update Gate and Reset Gate [13]. Long short-term memory (LSTM). The structure LSTM [11] reminds of same chain as on Fig. 3 but modules look and interact in a different way (see Fig. 3).

The key component of LSTM – is cell state. Cell state participates in several linear transformations. LSTM can erase information from cell state; this process is regulated by structures called gates. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation. LSTM-network gates: "input gate layer", "output gate layer" and "forget gate layer" [12]. "Forget gate

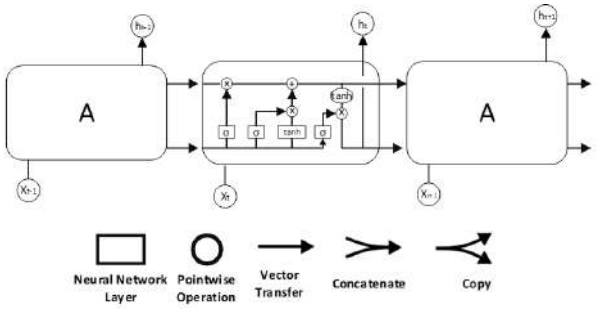

Figure 3. Recurrent LSTM network.

layer". The "forget gate layer" controls erasing or saving the cell state information based on the new input value and the value received from the previous iteration. The sigmoid layer returns numbers from zero to one which indicate what part of information should be passed on the network. Zero in this case means "do not miss anything", one – "skip all". "Input gate layer". "Input gate layer" determines when data should be written to the cell. The sigmoid layer, as before, returns numbers from zero to one based on the new input value and the value from the previous iteration. Now the sigmoid layer indicates what part information will be recorded, and the tanhlayer generates values that can be added to the cell state. As a result combination of new value and previous one will be recorded to the cell. When the data should be saved and to what extent and when they should be replaced with new ones and to what extent the neural network "decides itself" in learning process. "Output gate layer". "Output gate layer" determines what information we receive at the output. The output data will be based on the cell state and some gates will be applied to it. First, a sigmoid layer decides what information from the cell state we will output. The cell state values then pass through the tanhlayer to obtain values from the range -1 to 1 at the output and are multiplied with the output values of the sigmoid layer allowing only the required information to

be displayed. Gated Recurrent Unit (GRU). The structure GRU (See Fig. 4) as LSTM has some gates: "Update gate layer" and "forget gate layer", their purpose is the same as one.

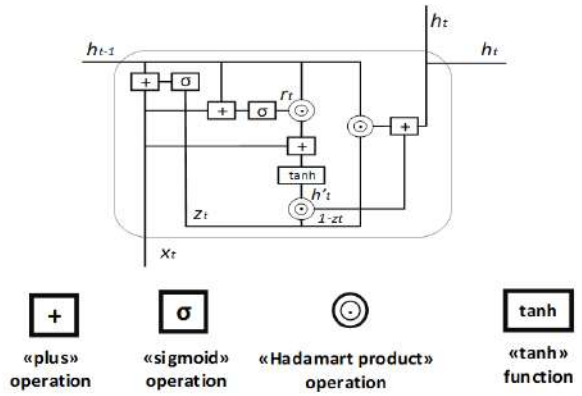

Figure 4. Recurrent GRU network.

IV. Neural network architecture.

The following neural network architecture was developed experimentally (see Fig. 5). The network consists of an input layer, three LSTM layers and one dense output layer with activation function Relu. The input data sets are 5100 in size (depth of immersion = 100). On the first LSTM layer there are 500 units and the other two have 100 units each. On the last layer there is 1 unit as we need only one output (forecast horizon = 1). After every LSTM layer the Dropout [14] is used, i.e. when training every training sample the network is rebuilt in such way that a certain set of units falls out (see Fig. 6). Developed neural network has the following view (see Fig. 7).

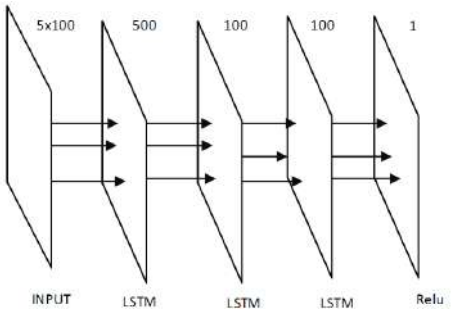

Figure 5. The architecture of proposed model.

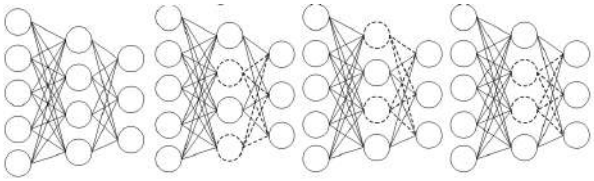

Figure 6. Dropout.

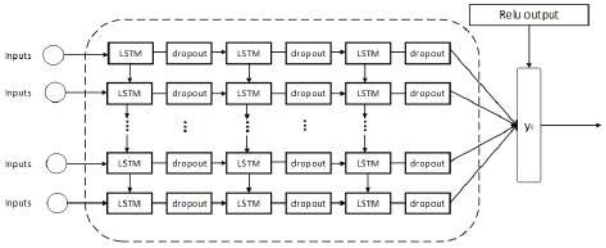

Figure 7. Neural network architecture.

V. Experiment

In the environment of Colab Laboratory [15] there was performed a software implementation of a neural network for future stock price prediction with LSTM using Python's Keras Library. Colab Laboratory is a free environment for Jupyter notebooks which requires no setting up and runs entirely in the cloud. Colab Laboratory allows you to write and execute code, as well as to get access powerful computing resources, which is an important advantage when working with resource-intensive machine learning algorithms. Keras [16] is one of the most powerful and easy-to-use Python's libraries for the development and evaluation of deep learning models covering the effective libraries of numerical computations Theano and TensorFlow. The advantage of this is mainly that it is possible to work with neural networks in a fairly simple way. Let's conduct experiments on neural network training (Tables IV- XII). As an optimizer we will use

Table IV
Experiments on network training for optimizer = «SGD», Simple RNN.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
50	0.0931	0.2512	0.0505	0.2245	0.0938	0.3038
100	0.00	0.0562	0.0053	0.0711	0.0039	0.0583
150	0.0041	0.0502	9.05e-4	0.0244	0.0004	0.0138
200	0.0026	0.0367	6.54e-4	0.0236	0.0020	0.0420
250	0.0019	0.0321	0.003	0.0537	0.0032	0.0530
300	0.0023	0.0356	4.99e-4	0.0176	0.0004	0.0148
350	0.0016	0.0306	3.31e-4	0.0147	0.0004	0.0162
400	0.0022	0.0351	1.85e-4	0.0116	0.0005	0.0170
450	0.0014	0.0275	6.88e-4	0.0215	0.0014	0.0286
500	0.0014	0.0274	3.06e-4	0.0141	0.0007	0.0223

a stochastic gradient descent optimizer (Tables IV- VI), which is traditional for neural networks training. Loss function (function of optimization estimation) – mean squared error. The metric (the function that is used to evaluate the model) - mean absolute error.

VI. Results and discussion

In this work the neural network model has been suggested in the area of stock prices forecast for pharmacies chain "36 and 6". The neural network architecture has been developed. Network consists of LSTM layers and a

Table V
EXPERIMENTS ON NETWORK TRAINING FOR OPTIMIZER = «SGD», GRU.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
...	...	...	...	...	...	...
250	0.0019	0.0321	0.0034	0.0537	0.0032	0.0530
275	0.0023	0.0371	9.29e-4	0.0259	0.0004	0.0160
300	0.0023	0.0356	4.99e-4	0.0176	0.0004	0.0148
325	0.0017	0.0301	5.09e-4	0.0196	0.0005	0.0170
350	0.0016	0.0306	3.31e-4	0.0147	0.0004	0.0162
375	0.0015	0.0293	2.68e-4	0.0139	0.0005	0.0168
400	0.0022	0.0351	1.85e-4	0.0116	0.0005	0.0170
425	0.0013	0.0275	5.68e-4	0.0191	0.0012	0.0228
450	0.0014	0.0275	6.88e-4	0.0215	0.0014	0.0286
475	0.0019	0.0322	1.40e-4	0.0079	0.0003	0.0155
500	0.0014	0.0274	3.06e-4	0.0141	0.0007	0.0223

Table VI
EXPERIMENTS ON NETWORK TRAINING FOR OPTIMIZER = «SGD», LSTM.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
150	0.0024	0.0347	1.58e-4	0.0090	0.0002	0.0118
175	0.1967	0.4219	0.0538	0.2313	0.0148	0.1039
200	0.0021	0.0318	1.72e-4	0.0099	0.0002	0.0128
225	0.0019	0.0303	2.24e-4	0.0119	0.0004	0.0182
250	0.0019	0.0297	1.41e-4	0.0084	0.0001	0.0107
275	0.0019	0.0299	1.90e-4	0.0105	0.0002	0.0144
300	0.0019	0.0295	2.33e-4	0.0122	0.0004	0.0189
325	0.0020	0.0310	1.65e-4	0.0094	0.0003	0.0129
350	0.0018	0.0297	2.28e-4	0.0119	0.0005	0.0199
375	0.0018	0.0291	1.31e-4	0.0080	0.0002	0.0108
400	0.0017	0.0279	2.16e-4	0.0118	0.0005	0.0199

It is recommended to use "RMSprop" optimizer for recurrent neural networks according to [11]

Table VII
EXPERIMENTS ON NETWORK TRAINING FOR OPTIMIZER = "RMSPROP", SIMPLE RNN.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
...	...	...	...	...	...	...
200	0.1967	0.4219	0.0538	0.231	0.0148	0.1039
250	0.0210	0.1201	0.0518	0.231	0.1297	0.3545
300	0.1967	0.4219	0.0538	0.231	0.0148	0.1039
350	0.0205	0.1180	0.0251	0.157	0.0852	0.2849
400	0.0094	0.0755	2.87e-4	0.013	0.0215	0.1319
450	0.1967	0.4219	0.0538	0.231	0.0148	0.1039
500	0.1967	0.4219	0.0538	0.231	0.03523	0.1877
600	0.1967	0.4219	0.0538	0.231	0.0148	0.1039
700	0.1967	0.4219	0.0538	0.231	0.0148	0.1039
800	0.0201	0.1163	0.0129	0.112	0.0615	0.2399
900	0.0196	0.1158	0.0299	0.172	0.0938	0.2997
1000	0.1967	0.4219	0.0538	0.231	0.0148	0.1039

Table VIII
EXPERIMENTS ON NETWORK TRAINING FOR OPTIMIZER = "RMSPROP", GRU.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
100	0.0012	0.0280	2.88e-5	0.0047	2.24e-5	0.0042
125	0.0011	0.0258	1.16e-5	0.0028	0.0002	0.0114
150	9.01e-4	0.0233	1.68e-4	0.0127	0.0005	0.0227
175	7.62e-4	0.0217	1.44e-4	0.0117	7.79e-5	0.0082
200	5.24e-4	0.0175	1.12e-5	0.0027	7.71e-5	0.0075
225	6.43e-4	0.0193	8.59e-5	0.0088	0.0004	0.0182
250	4.70e-4	0.0169	6.39e-4	0.0251	0.0018	0.0417
275	4.46e-4	0.0162	1.80e-5	0.0036	7.51e-5	0.0080
300	3.72e-4	0.0151	1.48e-5	0.0032	3.13e-5	0.0049
325	3.62e-4	0.0144	5.22e-5	0.0069	0.0001	0.0107
350	1.85e-4	0.0100	6.24e-6	0.0018	1.18e-5	0.0019
375	3.26e-4	0.0140	3.72e-4	0.0191	0.0007	0.0265
400	2.77e-4	0.0127	1.55e-4	0.0121	0.0006	0.0231

Table IX
EXPERIMENTS ON NETWORK TRAINING FOR OPTIMIZER = "RMSPROP", LSTM.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
100	6.026e-4	0.0185	1.487e-4	0.0119	0.0003	0.0220
125	8.697e-4	0.0228	1.723e-5	0.0034	0.0003	0.0173
150	3.978e-4	0.0146	4.024e-5	0.0057	0.0005	0.0256
175	6.145e-4	0.0189	8.031e-5	0.0086	5.27e-5	0.0066
200	7.348e-4	0.0208	1.654e-4	0.0125	0.0002	0.0103
225	5.222e-4	0.0179	9.030e-5	0.0090	0.0008	0.0259
250	4.442e-4	0.0166	1.405e-5	0.0032	6.09e-5	0.0069
275	4.244e-4	0.0149	1.830e-4	0.0133	0.0003	0.0170
300	3.782e-4	0.0146	0.0013	0.0341	0.0083	0.0861
325	4.915e-4	0.0176	7.736e-5	0.0076	0.0026	0.0423
350	3.653e-4	0.0139	4.351e-4	0.0207	0.0007	0.0256
375	0.1967	0.4219	0.0538	0.2313	0.0148	0.1039
400	0.1967	0.4219	0.0538	0.2313	0.0148	0.1039

Table X- XII shows the results using another common optimizer «adam». A good results are obtained for a LSTM neural network trained in 150 epochs, optimizer «adam». The best results are obtained for a neural network trained in 350 and 400 epochs.

Table X
EXPERIMENTS ON NETWORK TRAINING FOR OPTIMIZER = «ADAM», SIMPLE RNN.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
100	0.0209	0.119	0.0516	0.226	0.1292	0.3538
125	0.0212	0.119	0.0646	0.253	0.1491	0.3809
150	0.1967	0.421	0.0538	0.231	0.0148	0.1039
175	0.0201	0.117	0.0619	0.248	0.1424	0.3723
200	0.0017	0.036	5.32e-5	0.005	0.0012	0.029
250	0.1967	0.421	0.0538	0.231	0.0321	0.1468
300	0.1967	0.421	0.0538	0.231	0.0148	0.1039
325	0.0045	0.053	0.0038	0.057	0.0032	0.0497
350	0.1967	0.421	0.0538	0.231	0.0148	0.1039
400	0.1967	0.421	0.0538	0.231	0.0148	0.1039
450	0.1967	0.421	0.0538	0.231	0.0148	0.1039
475	0.0012	0.026	6.05e-5	0.006	0.0016	0.0331
500	0.1967	0.421	0.0538	0.233	0.0148	0.1039

Table XI
EXPERIMENTS ON NETWORK TRAINING FOR OPTIMIZER = «ADAM»,GRU.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
100	5.11e-4	0.0164	1.42e-5	0.0027	3.84e-5	0.0055
125	4.94e-4	0.0164	4.57e-5	0.0058	8.36e-5	0.0086
150	3.27e-4	0.0136	3.28e-5	0.0053	2.70e-5	0.0047
175	3.51e-4	0.0138	6.15e-5	0.0073	7.88e-5	0.0086
200	2.91e-4	0.0123	2.98e-5	0.0047	0.0148	0.1039
225	3.25e-4	0.0137	8.18e-6	0.0022	1.90e-5	0.0038
250	2.36e-4	0.0111	2.62e-5	0.0045	1.11e-5	0.0027
275	2.31e-4	0.0113	1.20e-5	0.0028	3.66e-5	0.0055
300	1.87e-4	0.0099	9.99e-6	0.0026	6.20e-5	0.0066
325	2.04e-4	0.0106	1.39e-5	0.0029	3.94e-5	0.0057
350	2.08e-4	0.0109	2.64e-5	0.0047	1.52e-5	0.0032
375	1.72e-4	0.0095	5.43e-6	0.0016	9.69e-6	0.0025
400	1.84e-4	0.0099	1.20e-5	0.0028	5.27e-6	0.0017

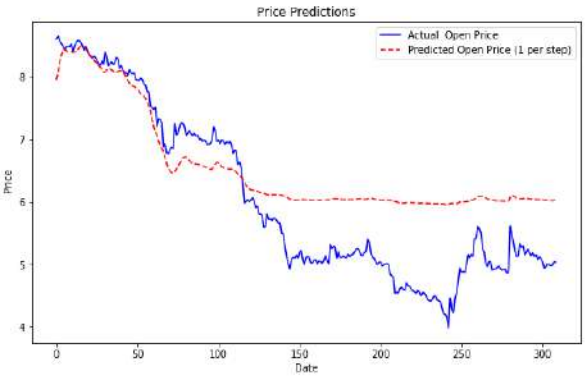

Figure 9. The results of the forecast on the test sample (P = 475, optimizer = "adam"). SimpleRNN.

Table XII
EXPERIMENTS ON NETWORK TRAINING FOR OPTIMIZER = «ADAM», LSTM.

Ep	Training set		Validation set		Test set	
	MSE	***MAE***	***MSE***	***MAE***	***MSE***	***MAE***
100	7.98e-4	0.0215	3.71e-5	0.0041	3.69e-5	0.0041
125	4.73e-4	0.0159	1.19e-4	0.0099	0.0305	0.1746
150	5.43e-4	0.0174	2.58e-5	0.0036	3.44e-5	0.0043
175	4.08e-4	0.0153	2.17e-5	0.0032	4.75e-5	0.0058
200	3.54e-4	0.0141	2.76e-4	0.0162	0.0003	0.0177
225	3.27e-4	0.0134	1.44e-5	0.0028	5.05e-5	0.0061
250	2.56e-4	0.0117	5.37e-5	0.0065	4.58e-5	0.0061
275	0.0019	0.0306	2.21e-4	0.0118	0.0003	0.0165
300	2.81e-4	0.0126	2.51e-5	0.0041	2.93e-5	0.0048
325	2.30e-4	0.0110	9.75e-6	0.0024	2.61e-5	0.0041
350	1.85e-4	0.0096	8.27e-6	0.0022	3.34e-5	0.0049
375	2.03e-4	0.0102	1.61e-4	0.0124	0.0003	0.0166
400	1.75e-4	0.0098	8.63e-6	0.0022	1.66e-5	0.0033

Figure 10. The results of prediction on the test sample (Steps = 150, optimizer = "RMSprop").GRU.

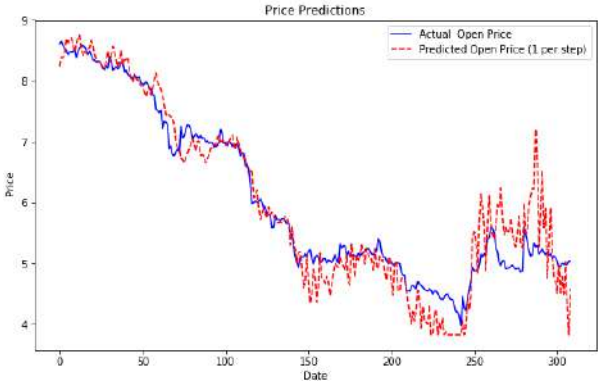

Figure 8. The results of prediction on the test sample (Steps = 150, optimizer = "SGD"). SimpleRNN.

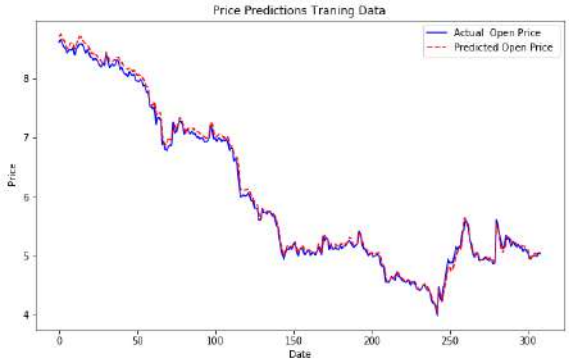

Figure 11. Forecast results on the test sample (Steps = 250, optimizer = "adam"). GRU.

Figure 12. Forecast results on a sample of ten (Steps = 175, optimizer = "RMSprop"). LSTM

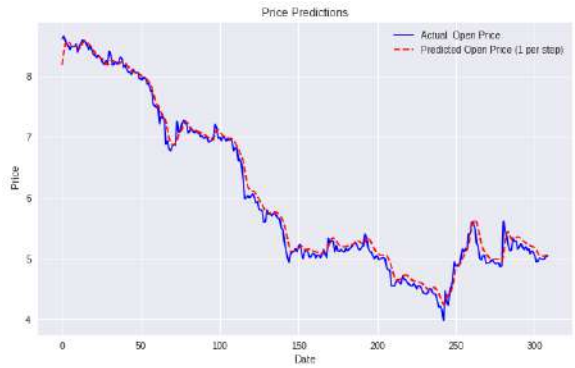

Figure 13. The results of prediction on the test sample (Steps = 150, optimizer = "adam").LSTM.

Dense layer with the Relu activation function. Dropout is used to solve the problem of retraining. Experimental studies have shown that the best results are achieved using the optimizer "adam". Error on a test set is MSE = 1.664e-05, MAE = 0.0033. This error makes it possible to forecast the price dynamics.

References

[1] Iacomin R.: Stock market prediction. In: 19th International Conference on System Theory, Control and Computing (ICSTCC), Cheile Gradistei, pp. 200-205. (2015), DOI: 10.1109/ICSTCC.2015.7321293.

[2] Shai, Shalev-Shwartz, Shai, Ben-David: Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press, 32 Avenue of the Americas, New York, NY 10013-2473, USA, 416p. (2014)

[3] Schmidhuber J.: Deep Learning in Neural Networks: an Overview. Neural Networks. Vol. 1. pp. 85–117. (2015.) DOI: 10.1016/j.neunet.2014.09.003

[4] Brockwell, Peter J., Davis, Richard A.: Introduction to Time Series and Forecasting. Second Edition. Springer-Verlag New York, Inc., 175 Fifth Avenue, New York, NY 10010, USA, 449 p. (2002)

[5] Sergii, K., Yurii, S., Tatyana, V., Natalia, A.: Feature Selection for Time-Series Prediction in Case of Undetermined Estimation. In: Samsonovich A., Klimov V., Rybina G. (eds.) Biologically Inspired Cognitive Architectures (BICA) for Young Scientists, Advances in Intelligent Systems and Computing, vol. 449, pp. 85-97. Springer, Cham (2016).

[6] Stock quotes, https://www.finam.ru/, last accessed 2019/03/29.

[7] Vargas, R., Ruiz, L.: Deep learning: previous and present applications. Journal of awareness, 2(Special 3), 11-20 (2018).

[8] Chen Y., Lin Z., Zhao X., Wang G., and Gu Y.: Deep Learning-Based Classification of Hyperspectral Data. Selected Topics in Applied Earth Observations and Remote Sensing. IEEE Journal of. 7, 2094-2107(2014).

[9] Bao, W, Yue, J, Rao, Y.: A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS One 12(7), 1-24 (2017).

[10] Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05), vol. 1, pp. 539–546. IEEE, (2005).

[11] Manaswi, N.K.: Regression to MLP in Keras. In: Deep Learning with Applications Using Python, pp. 68-89. Apress, Berkeley, CARecurrent Neural Networks (2018).

[12] Vasyaeva T., Martynenko T., Khmilovyi S., Andrievskaya N. (2019) Stock Prices Forecasting with LSTM Networks. In: Kuznetsov S., Panov A. (eds) Artificial Intelligence. RCAI 2019. Communications in Computer and Information Science, vol 1093. Springer, Cham

[13] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Gated feedback recurrent neural networks. arXiv preprint arXiv:1502.02367, 2015

[14] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958. (2014).

[15] Google Colaboratory, https://colab.research.google.com/notebooks/welcome.ipynb, last accessed 2019/02/11.

[16] Keras: The Python Deep Learning library, https://Keras.io/, last accessed 2019/03/10.

Прогнозирование динамики изменения цен на акции с помощью рекуррентных нейронных сетей.

Васяева Т.А., Мартыненко Т.В.,
Хмелевой С.В., Андриевская Н.К.

В статье рассмотрено применение рекуррентных нейронных сетей для прогнозирования цен акций сети аптек «Эбиб». Предметом исследования данной работы являются методы анализа фондовых бирж. Показано получение котировок акций, описан формат полученной информации. Данные представляют собой многомерный временной ряд, который содержит 5 каналов: цена открытия, максимальная цена, минимальная цена, цена закрытия, объем торгов. Рассмотрено формирование обучающей выборки «скользящим окном» в задачах машинного обучения применительно к многомерным временным рядам. Рекуррентные нейронные сети трудно обучать из-за проблемы исчезающего градиента. Показано применение GRU и LSTM сетей для решения данной проблемы. Предложена архитектура нейронной сети. Разработана и обучена нейронная сеть для прогнозирования биржевой торговли с использованием языка программирования Python, а также библиотеки Keras. Проведены эксперименты по настройке основных параметров.

Received 14.12.2019