DOI: 10.20535/kpisn.2021.3.230989
UDC 519.246.8+519.216.3+004.032.26

V.V. Romanuke*
Polish Naval Academy, Gdynia, Poland
*corresponding author: romanukevadimv@gmail.com

OPTIMIZATION OF LSTM NETWORKS FOR TIME SERIES FORECASTING

Received: 9 May 2021. Accepted: 7 Dec. 2021.

Background. LSTM neural networks are a very promising means to develop time series analysis and forecasting. However, as well as neural networks for other fields and applications, LSTM networks have a lot of architecture versions, training parameters, and hyperparameters, whose inappropriate selection may lead to unacceptably poor performance (poor or badly unreliable forecasts). Thus, optimization of LSTM networks is still an open question.
Objective. The goal is to ascertain whether the best forecasting accuracy is achieved at such a number of LSTM layer neurons, which can be determined by the time series lag.
Methods. To achieve the said goal, a set of benchmark time series for testing the forecasting accuracy is presented. Then, a set-up of the computational study for various versions of the LSTM network is defined. Finally, the computational study results are clearly visualized and discussed.
Results. Time series with a linear trend are forecasted worst, whereas defining the LSTM layer size by the lag in a time series does not help much. The best-forecasted are time series with only repeated random subsequences, or seasonality, or exponential rising. Compared to the single LSTM layer network, the forecasting accuracy is improved by 15 % to 19 % by applying the two LSTM layers network.
Conclusions. The approximately best forecasting accuracy may be expectedly achieved by setting the number of LSTM layer neurons at the time series lag. However, the best forecasting accuracy cannot be guaranteed. LSTM networks for time series forecasting can be optimized by using only two LSTM layers whose size is set at the time series lag. Some discrepancy is still acceptable, though. The size of the second LSTM layer should not be less than the size of the first layer.

Keywords: time series forecasting; LSTM network; LSTM layer size; forecasting accuracy; root-mean-square error; maximum absolute error.

Introduction

Time series analysis and forecasting is an important field of methods to control and predict processes comprising sequences of data [1], [2]. It has both deep theoretical and practical applications in industrial, socio-economic, ecological, entertainment, and scientific branches [2], [3]. A common example of a data sequence to be forecasted is a set of successive equally spaced points in time, at which practically influential values are registered and thus are studied [1], [4].

The forecasting accuracy strongly depends on a model used to generate forecasts. Besides, it depends on the forecasting horizon. As the horizon is extended, the accuracy decreases and forecasts become less reliable. It has been recently reported [5], [6] that, along with statistical forecasting methods considering averages, regression factors, and scedasticity of data (like ARIMA-based models, GARCH-based models), recurrent neural networks based on long short-term memory (LSTM) are capable to achieve the same accuracy or even better. Therefore, LSTM networks are a very promising means to develop time series analysis and forecasting. However, as well as neural networks for other fields and applications, LSTM networks have a lot of architecture versions, training parameters, and hyperparameters, whose inappropriate selection may lead to unacceptably poor performance (poor or badly unreliable forecasts) [5], [7], [8]. Thus, optimization of LSTM networks is still an open question.

Рекомендуємо цитувати цю статтю так: V.V. Romanuke, “Optimization of LSTM networks for time series forecasting”, *Наукові вісті КПІ*, № 3, с. 14–25, 2021, doi: 10.20535/kpisn.2021.3.230989.

Please cite this article as: V.V. Romanuke, “Optimization of LSTM networks for time series forecasting”. *KPI Science News*, no. 3, pp. 14–25, 2021, doi: 10.20535/kpisn.2021.3.230989.

Problem statement

The most important characteristic of an LSTM network is its architecture. In particular, it includes the number of LSTM layers and the number of neurons in each layer. It is believed that the optimal number of LSTM layer neurons somehow correlates with time series lags [6], [7], [9]. The lag is an approximate length of a time series part, which quasi-periodically recurs. Lags can be found (or estimated) by the autocorrelation function (ACF) of the time series [9]. Therefore, the goal is to ascertain whether the best forecasting accuracy is achieved at such a number of LSTM layer neurons, which can be determined by the lag. For this, a set of benchmark time series for testing the forecasting accuracy will be presented first. Then, a set-up of the computational study for various versions of the LSTM network is defined. Finally, the computational study results are to be clearly visualized and discussed, whereupon the corresponding conclusions on optimization of LSTM networks for time series forecasting will be made.

Time series benchmarking dataset

Denote by T the amount of a time series data. The benchmarking dataset is based on 12 patterns of random-like sequences with repeatability, where every sequence is a stack of 6, 7, or 8 identical randomly-structured subsequences. These sequences are denoted by $\{r_g(t)\}_{g=1}^{12}$. Every sequence is generated by using pseudorandom numbers drawn from the standard normal distribution (with zero mean and unit variance) [10], [11] by, without losing generality, $t = \overline{1, T}$.

An initial set of benchmark time series is generated as follows. First, vectors $\{\Theta_l(T)\}_{l=1}^{30}$ of T pseudorandom numbers used to simulate noise and volatility are generated. Then, a set $\{a_h > 0\}_{h=1}^{6}$ of adjustable coefficients and factor $\upsilon > 0$ indicating an oscillation frequency are defined for all the 12 patterns [12].

In the simplest case, a time series pattern without additional properties is

$$y_1(t) = [a_1 + 0.25\Theta_1(T)]r_1(t) + a_2\Theta_2(T). \quad (1)$$

A time series pattern with a linear trend is

$$y_2(t) = [a_1 + 0.25\Theta_3(T)]r_2(t) + a_2\Theta_4(T) + a_3 t, \quad (2)$$

and a time series pattern with seasonality is

$$y_3(t) = [a_1 + 0.25\Theta_5(T)]r_3(t) + a_2\Theta_6(T) + [a_4 + 0.25\Theta_7(T)]a_5\cos(\upsilon t)\,. \quad (3)$$

Three patterns (1)—(3) are used in various combinations to form the remaining nine patterns including exponential extinction and rising properties. Thus, a time series pattern with a linear trend and seasonality is

$$y_4(t) = [a_1 + 0.25\Theta_8(T)]r_4(t) + a_2\Theta_9(T) + a_3 t + [a_4 + 0.25\Theta_{10}(T)]a_5\cos(\upsilon t)\,. \quad (4)$$

Time series patterns with exponential extinction and rising are

$$y_5(t) = [a_1 + 0.25\Theta_{11}(T)]r_5(t)e^{-a_6 t} + a_2\Theta_{12}(T) \quad (5)$$

and

$$y_6(t) = [a_1 + 0.25\Theta_{13}(T)]r_6(t)e^{a_6 t} + a_2\Theta_{14}(T), \quad (6)$$

respectively. Next, a time series pattern with a linear trend with exponential extinction is

$$y_7(t) = [a_1 + 0.25\Theta_{15}(T)]r_7(t)e^{-a_6 t} + a_2\Theta_{16}(T) + a_3 t. \quad (7)$$

If the seasonality substitutes the linear trend, another pattern is generated (seasonality with exponential extinction):

$$y_8(t) = [a_1 + 0.25\Theta_{17}(T)]r_8(t)e^{-a_6 t} + a_2\Theta_{18}(T) + [a_4 + 0.25\Theta_{19}(T)]a_5\cos(\upsilon t)e^{-a_6 t}. \quad (8)$$

A time series pattern with a linear trend and seasonality with exponential extinction is

$$y_9(t) = [a_1 + 0.25\Theta_{20}(T)]r_9(t)e^{-a_6 t} + a_2\Theta_{21}(T) + a_3 t + [a_4 + 0.25\Theta_{22}(T)]a_5\cos(\upsilon t)e^{-a_6 t}. \quad (9)$$

The final three patterns are similar to patterns (7)—(9), where only exponential rising is embedded instead of the extinction:

$$y_{10}(t) = [a_1 + 0.25\Theta_{23}(T)]r_{10}(t)e^{a_6 t} + a_2\Theta_{24}(T) + a_3 t, \quad (10)$$

$$y_{11}(t) = [a_1 + 0.25\Theta_{25}(T)]r_{11}(t)e^{a_6 t} + a_2\Theta_{26}(T) + [a_4 + 0.25\Theta_{27}(T)]a_5\cos(\upsilon t)e^{a_6 t}, \quad (11)$$

$$y_{12}(t) = [a_1 + 0.25\Theta_{28}(T)]r_{12}(t)e^{a_6 t} + a_2\Theta_{29}(T) + a_3 t + [a_4 + 0.25\Theta_{30}(T)]a_5\cos(\upsilon t)e^{a_6 t}. \quad (12)$$

The initial time series benchmarking dataset is generated by [12]

$$a_1 = 2,\ a_2 = 0.175,\ a_3 = 0.01,\ a_4 = 5,\ a_5 = 0.18,\ \upsilon = 0.02,\ a_6 = 0.0005,\ T = 1680.$$

Then the time series is equidistantly downsampled so that 168 time points remain. These points are smoothed. For each of patterns (1)—(12), 200 series are generated. For each of those 2400 series, ARIMA forecasts [12] are made at $t = \overline{113, 168}$ (i. e, the forecast length is one third of the available data). The forecasting accuracy is estimated by the corresponding root-mean-square error (RMSE) and the maximum absolute error (MaxAE) [4], [13], [14] as follows. If

$$\{\tilde{y}(t)\}_{t=113}^{168} \tag{13}$$

are forecasted data, they are normalized with respect to the initial data:

$$\tilde{u}(t) = \frac{\tilde{y}(t) - \min\limits_{k=113,\,168} y(t)}{\max\limits_{k=113,\,168} y(t) - \min\limits_{k=113,\,168} y(t)}$$

$$\text{by } t = \overline{113, 168}. \tag{14}$$

Test data

$$\{y(t)\}_{t=113}^{168} \tag{15}$$

are normalized as well:

$$u(t) = \frac{y(t) - \min\limits_{k=113,\,168} y(t)}{\max\limits_{k=113,\,168} y(t) - \min\limits_{k=113,\,168} y(t)}$$

$$\text{by } t = \overline{113, 168}. \tag{16}$$

Then the RMSE registering information about the averaged errors is calculated as

$$\rho_{\text{RMSE}} = \sqrt{\frac{1}{56} \sum_{t=113}^{168} [u(t) - \tilde{u}(t)]^2}, \tag{17}$$

and the MaxAE registering information about the worst errors [12] is calculated as

$$\rho_{\text{MaxAE}} = \max_{t=113,\,168} |u(t) - \tilde{u}(t)|. \tag{18}$$

The 50 time series which are forecasted the worst are extracted for each pattern. Their respective RMSEs are sorted in descending order, so each series is tagged to its number $z = \overline{1, 50}$ ($z = 1$ corresponds to the maximal RMSE). The time series starting value $y_g(1)$, which can be heavily distorted with respect to the next values $y_g(2)$, $y_g(3)$, ..., as a consequence of the smoothing, is modified as follows:

$$y_g^{(\text{obs})}(1) = y_g(1),$$

$$y_g(1) = y_g(2) \cdot (0.2\xi + 0.9), \tag{19}$$

where ξ is a pseudorandom scalar drawn from the standard uniform distribution on the open interval (0; 1). Thus, $y_g(1)$ becomes equal $y_g(2)$ multiplied by a factor between 0.9 and 1.1 (after rounding; speaking more precisely, the minimal value of the factor is greater than 0.9 by infinitesimal, and the maximal value of the factor is less than 1.1 by infinitesimal).

Graphical examples of four benchmark time series per pattern (out of those 50 ones which are forecasted the worst) are presented in Fig. 1. The start of forecasting is marked as vertical line. The line separates the two thirds of the time series from its one third which is the part to be forecasted. Fig. 1 allows seeing how patterns (1)—(12) factually appear representing the most important properties of a time series (trend, seasonality, exponential extinction and rising). It is worth noting that the time series starting value modification by (19) does not always help. Indeed, the starting value $y_1(1)$ for the first benchmark time series (the top left corner subplot) is clearly seen to be an outlier. Similar “jumps” can be seen in other patterns, although they are far less apparent.

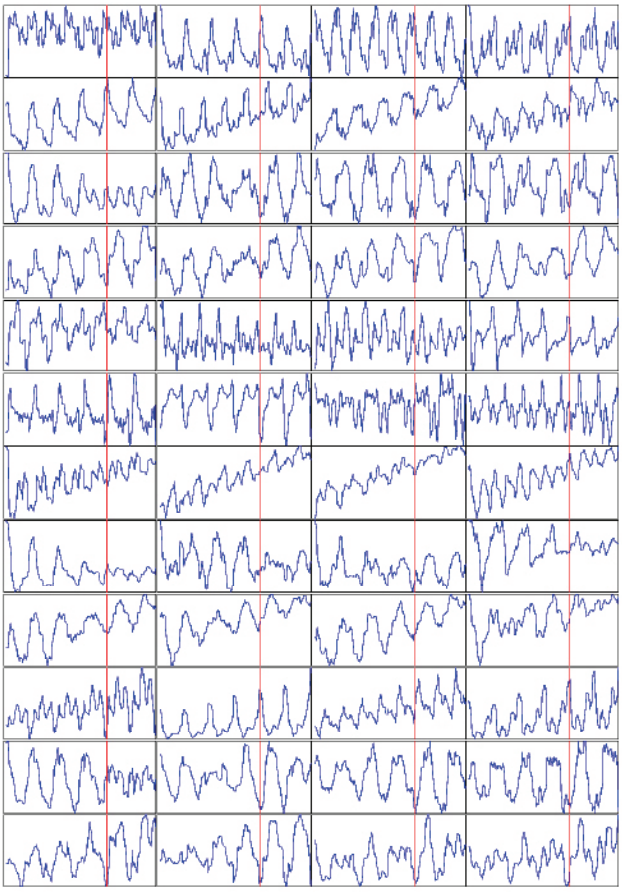


Fig. 1. Four benchmark time series per pattern

Computational study set-up

First, the time series

$$\{y_g(t)\}_{t=1}^{112} \tag{20}$$

is approximated by a linear trend model

$$\bar{y}_{\text{trend}}(t) = \beta_0 + \beta_1 t. \tag{21}$$

Then the ACF of the sequence

$$\{y_g(t) - \bar{y}_{\text{trend}}(t)\}_{t=1}^{112} \tag{22}$$

is found (e. g., see [9]). A set of all different lags in sequence (22) are determined by this ACF. Denote this set by P_{found}. Because every pattern

$$\{y_g(t)\}_{t=1}^{168} \tag{23}$$

is a stack of 6, 7, or 8 identical randomly-structured subsequences, the length of the subsequence is 28, 24, or 21, respectively. Therefore, the default set of lags is

$$P_{\text{def}} = \{21, 24, 28\}. \tag{24}$$

Sets P_{found} and (24) can coincide, or can have only one or two mutual elements, or their intersection can be even empty. This is why their union $P_{\text{def}} \cup P_{\text{found}}$ is considered further. Let this lag union be called consistent.

The default set of neurons in a LSTM layer is

$$H = \{h_k\}_{k=1}^{19} = \{20 + 10\cdot(k-1)\}_{k=1}^{19}. \tag{25}$$

Since a correlation between the optimal number of LSTM layer neurons and lags is to be ascertained, set (25) is supplemented with the sets of lags:

$$H_* = H \cup P_{\text{def}} \cup P_{\text{found}}. \tag{26}$$

If

$$|H_*| > |H| + |P_{\text{def}}| = |H| + 3 \tag{27}$$

then the largest $|H_*| - |H| - 3$ values in set H are deleted by

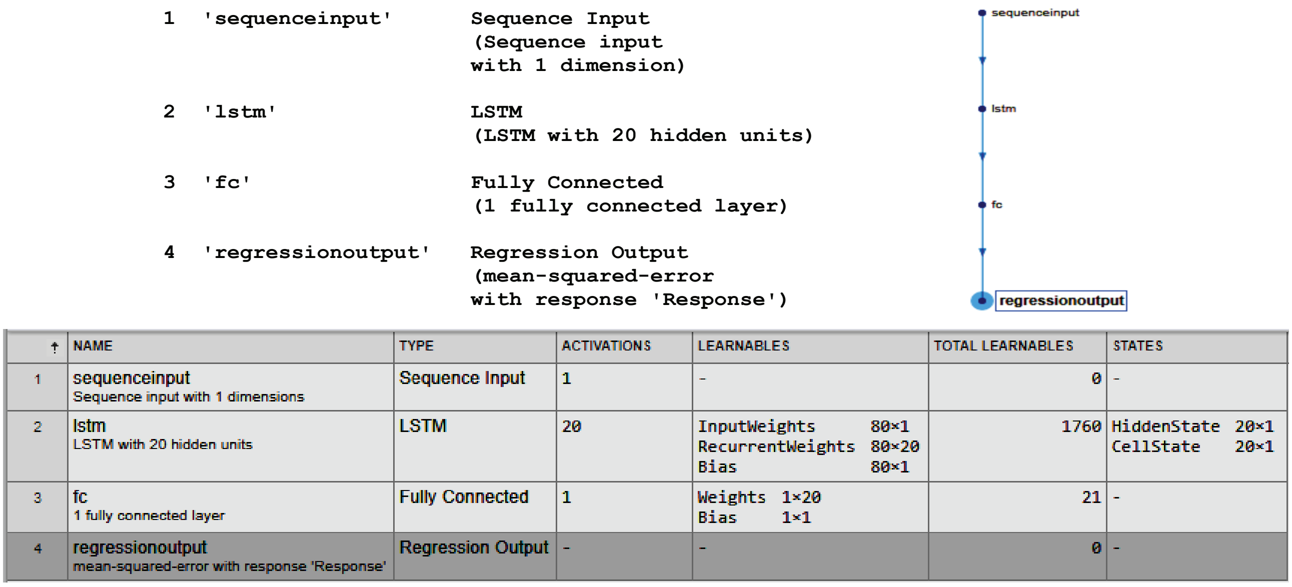
$$\bar{H} = \{20 + 10\cdot(k-1)\}_{k=1}^{22-|H_*|+|H|}, \tag{28}$$

whereupon

$$H_{**} = \bar{H} \cup P_{\text{def}} \cup P_{\text{found}}; \tag{29}$$

otherwise $H_{**} = H_*$.

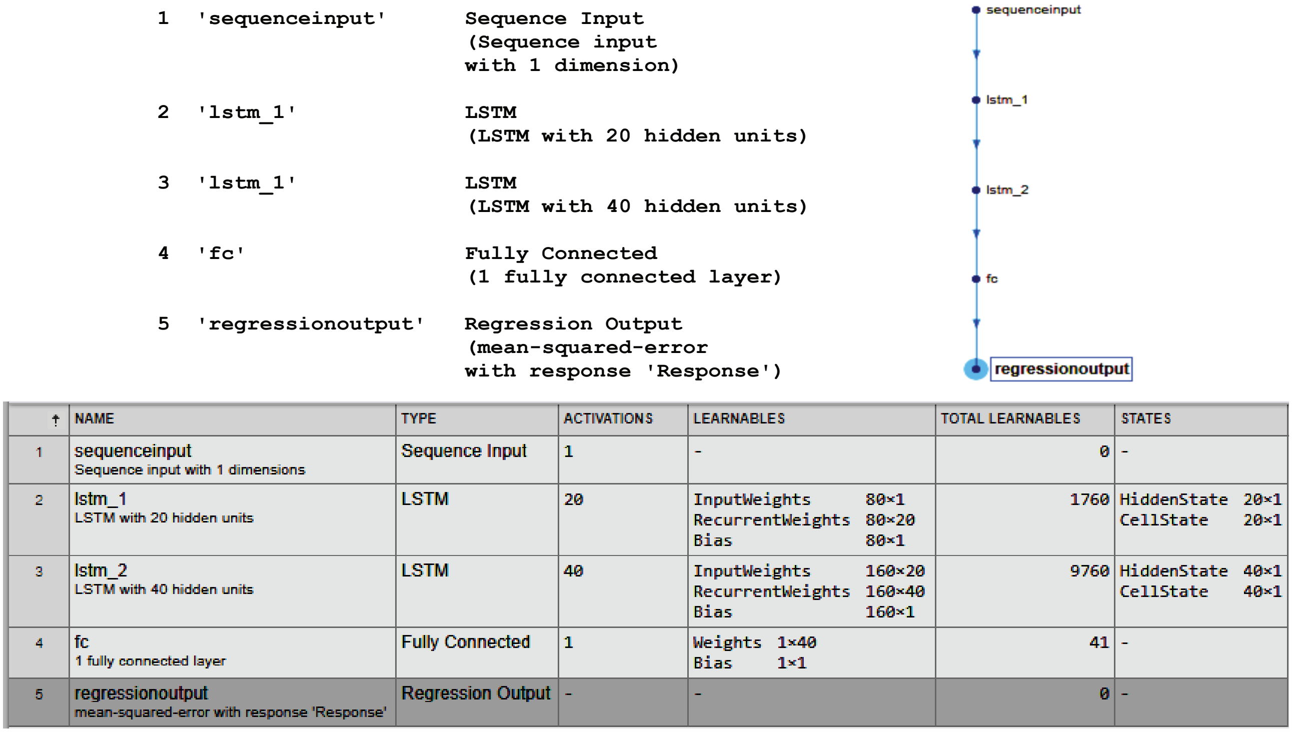
Two versions of the LSTM network are to be studied: with a single LSTM layer (Fig. 2) and two LSTM layers (Fig. 3), where the training parameters are set at values allowing to achieve distinguishable results. So, these values are not optimal with respect to the network performance but they are optimal (appropriate) with respect to the network operation speed and the performance distinguishability. The number of epochs is 300, the starting learning rate is 0.1, the learning rate drop factor is 0.995, and the learning rate drop period is 25 epochs [5], [15], [16].



```
1   'sequenceinput'        Sequence Input
                           (Sequence input
                           with 1 dimension)

2   'lstm'                 LSTM
                           (LSTM with 20 hidden units)

3   'fc'                   Fully Connected
                           (1 fully connected layer)

4   'regressionoutput'     Regression Output
                           (mean-squared-error
                           with response 'Response')
```

↑	NAME	TYPE	ACTIVATIONS	LEARNABLES	TOTAL LEARNABLES	STATES
1	sequenceinput Sequence input with 1 dimensions	Sequence Input	1	-	0	-
2	lstm LSTM with 20 hidden units	LSTM	20	InputWeights 80×1 RecurrentWeights 80×20 Bias 80×1	1760	HiddenState 20×1 CellState 20×1
3	fc 1 fully connected layer	Fully Connected	1	Weights 1×20 Bias 1×1	21	-
4	regressionoutput mean-squared-error with response 'Response'	Regression Output	-	-	0	-

Fig. 2. The LSTM network architecture with a single LSTM layer (an example where $h_1 = 20$)



```
1   'sequenceinput'       Sequence Input
                          (Sequence input
                          with 1 dimension)

2   'lstm_1'              LSTM
                          (LSTM with 20 hidden units)

3   'lstm_1'              LSTM
                          (LSTM with 40 hidden units)

4   'fc'                  Fully Connected
                          (1 fully connected layer)

5   'regressionoutput'    Regression Output
                          (mean-squared-error
                          with response 'Response')
```

↑	NAME	TYPE	ACTIVATIONS	LEARNABLES	TOTAL LEARNABLES	STATES
1	sequenceinput Sequence input with 1 dimensions	Sequence Input	1	-	0	-
2	lstm_1 LSTM with 20 hidden units	LSTM	20	InputWeights 80×1 RecurrentWeights 80×20 Bias 80×1	1760	HiddenState 20×1 CellState 20×1
3	lstm_2 LSTM with 40 hidden units	LSTM	40	InputWeights 160×20 RecurrentWeights 160×40 Bias 160×1	9760	HiddenState 40×1 CellState 40×1
4	fc 1 fully connected layer	Fully Connected	1	Weights 1×40 Bias 1×1	41	-
5	regressionoutput mean-squared-error with response 'Response'	Regression Output	-	-	0	-

Fig. 3. The LSTM network architecture with two LSTM layers (an example where the second layer is twice the size of the first one)

Each LSTM layer has its own set of neurons within set H_{**}. Re-denote this set by

$$H_{**} = \{h_{**}(g, z, k)\}_{k=1}^{22}, \tag{30}$$

where $h_{**}(g, z, k)$ is the k-th number (index) of neurons for the g-th pattern and z-th time series instance, $k = \overline{1, 22}$, $g = \overline{1, 12}$, $z = \overline{1, 50}$. Whichever number of LSTM layers is, regardless of index k, denote by

$$P(g, z) = P_{\text{def}} \cup P_{\text{found}} = \{p_l(g, z)\}_{l=1}^{L(g, z)} \tag{31}$$

a set of consistent lags for the g-th pattern and z-th time series instance, where $L(g, z) = |P(g, z)|$ is a number of consistent lags.

Results

For each triple of g, z, k, RMSE (17) by (13)—(16) for (20)—(23) and (24)—(31) is calculated for the case of a single LSTM layer (Fig. 2). Then, for set

$$\{\rho_{\text{RMSE}}(g, z, k)\}_{k=1}^{22} \tag{32}$$

a set

$$K_{\text{RMSE}}^{*} = \arg \min_{k=\overline{1, 22}} \rho_{\text{RMSE}}(g, z, k) \tag{33}$$

of indices at which RMSEs (32) are minimal is found as

$$K_{\text{RMSE}}^{*} = \{k_{\text{RMSE}}^{*(j)}\}_{j=1}^{|K_{\text{RMSE}}^{*}|}. \tag{34}$$

With set (34), integer $h_{**}(g, z, k_{\text{RMSE}}^{*(1)})$ is the minimal number of neurons in the LSTM layer, at which the RMSE is minimized. It is compared to the consistent lags: integer

$$\lambda_{\text{RMSE}}^{*}(g, z) = \min_{l=\overline{1, L(g, z)}} \left| h_{**}(g, z, k_{\text{RMSE}}^{*(1)}) - p_l(g, z) \right| \tag{35}$$

is the shortest distance (in neurons) between the LSTM layer size and consistent lags for the g-th pattern and z-th time series instance.

The number of instances when distance (35) is zero or, in other terms,

$$h_{**}(g, z, k_{\text{RMSE}}^{*(1)}) \in P(g, z) \tag{36}$$

is summed up along $z = \overline{1, 50}$. (Fig. 4). Besides, the average distance for the g-th pattern

$$\tilde{\lambda}_{\text{RMSE}}^{*}(g) = \frac{1}{50} \cdot \sum_{z=1}^{50} \lambda_{\text{RMSE}}^{*}(g, z) \tag{37}$$

is calculated. Averages (37) are compared to

$$\bar{\lambda}(g) =$$

$$\frac{1}{50}\cdot\sum_{z=1}^{50}\left|\frac{1}{22}\cdot\sum_{k=1}^{22}h_{**}(g, z, k)-\frac{1}{L(g, z)}\cdot\sum_{l=1}^{L(g, z)}p_l(g, z)\right|. \quad (38)$$

The difference

$$\delta_{\text{RMSE}}(g) = \bar{\lambda}(g) - \tilde{\lambda}^{*}_{\text{RMSE}}(g) \quad (39)$$

is shown in Fig. 5. Values (38) can be thought of as “averaged theoretical averages”, whereas (37) are the averages of real distances (35).

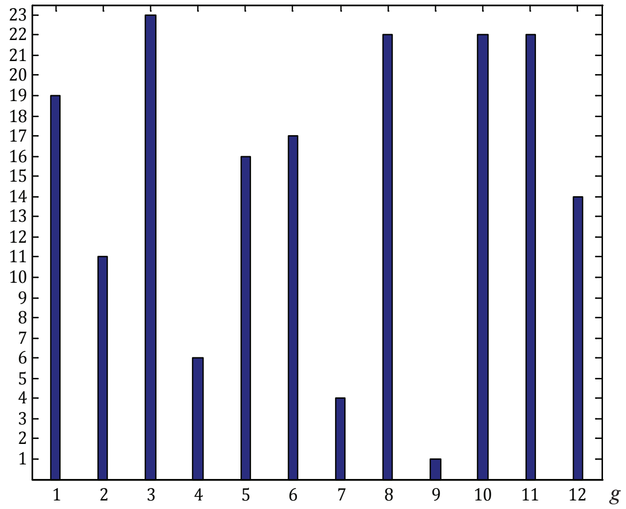


Fig. 4. The number of the LSTM-layer-size and consistent-lag coincidences by (36) in the case of a single LSTM layer (the possible maximum of this number is 50)

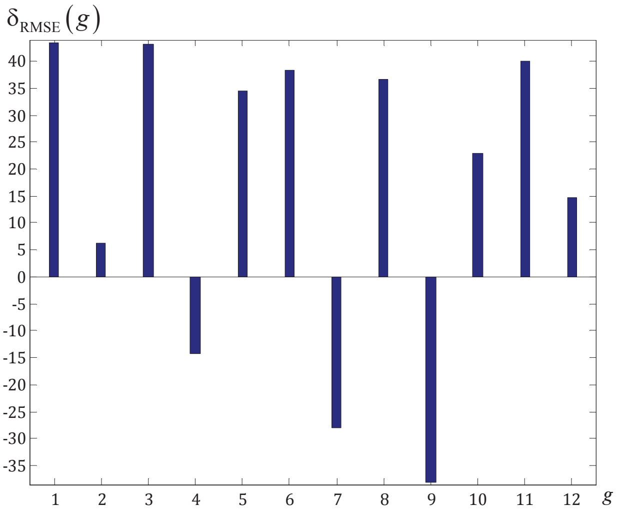


Fig. 5. Difference (39) in the case of a single LSTM layer

For each triple of g, z, k, MaxAE (18) by (13)–(16) for (20)–(23) and (24)–(31) is calculated for the case of a single LSTM layer (Fig. 2) likewise: for set

$$\{\rho_{\text{MaxAE}}(g, z, k)\}_{k=1}^{22} \quad (40)$$

a set

$$K^{*}_{\text{MaxAE}} = \arg\min_{k=\overline{1, 22}} \rho_{\text{MaxAE}}(g, z, k) \quad (41)$$

of indices at which MaxAEs (40) are minimal is found as

$$K^{*}_{\text{MaxAE}} = \{k^{*(j)}_{\text{MaxAE}}\}_{j=1}^{\left|K^{*}_{\text{MaxAE}}\right|}. \quad (42)$$

With set (42), integer $h_{**}(g, z, k^{*(1)}_{\text{MaxAE}})$ is the minimal number of neurons in the LSTM layer, at which the MaxAE is minimized. It is compared to the consistent lags: integer

$$\lambda^{*}_{\text{MaxAE}}(g, z) =$$

$$\min_{l=\overline{1, L(g, z)}}\left|h_{**}(g, z, k^{*(1)}_{\text{MaxAE}}) - p_l(g, z)\right| \quad (43)$$

is the shortest distance (in neurons) between the LSTM layer size and consistent lags for the g-th pattern and z-th time series instance.

Again, the number of instances when distance (43) is zero or, in other terms,

$$h_{**}(g, z, k^{*(1)}_{\text{MaxAE}}) \in P(g, z) \quad (44)$$

is summed up along $z = \overline{1, 50}$ (Fig. 6). Besides, the average distance for the g-th pattern

$$\tilde{\lambda}^{*}_{\text{MaxAE}}(g) = \frac{1}{50}\cdot\sum_{z=1}^{50}\lambda^{*}_{\text{MaxAE}}(g, z) \quad (45)$$

is calculated. Averages (45) are compared to (38): the respective difference

$$\delta_{\text{MaxAE}}(g) = \bar{\lambda}(g) - \tilde{\lambda}^{*}_{\text{MaxAE}}(g) \quad (46)$$

between the “averaged theoretical averages” and (45) is shown in Fig. 7.

Figs. 4–7 show that the best forecasting accuracy can hardly be achieved by defining a number of LSTM layer neurons as the lag in a time series generated by patterns (4), (7), (9). Indeed, there is only one instance (see Fig. 4) of the time series with a linear trend and seasonality with exponential extinction ($g = 9$, see the ninth subplot row in Fig. 1) at which the RMSE has been obtained minimal by setting the number of neurons in the LSTM layer at 35 that is the lag in the instance. This is the 17-th instance, where, by the way,

$$P(9, 17) = \{p_l(9, 17)\}_{l=1}^{6} = \{21, 24, 26, 28, 33, 35\}.$$

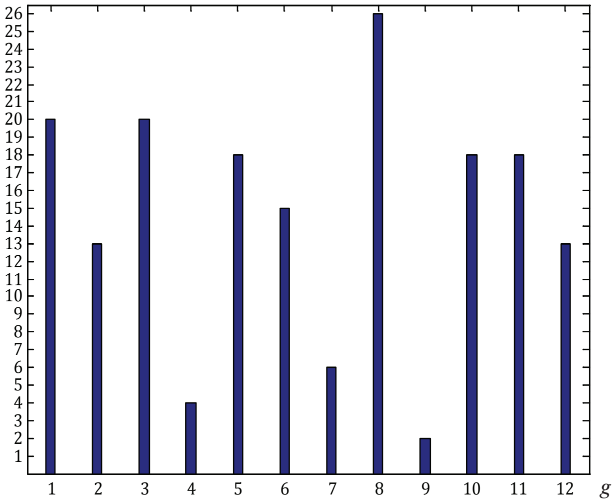


Fig. 6. The number of the LSTM-layer-size and consistent-lag coincidences by (44) in the case of a single LSTM layer (the possible maximum of this number is 50)

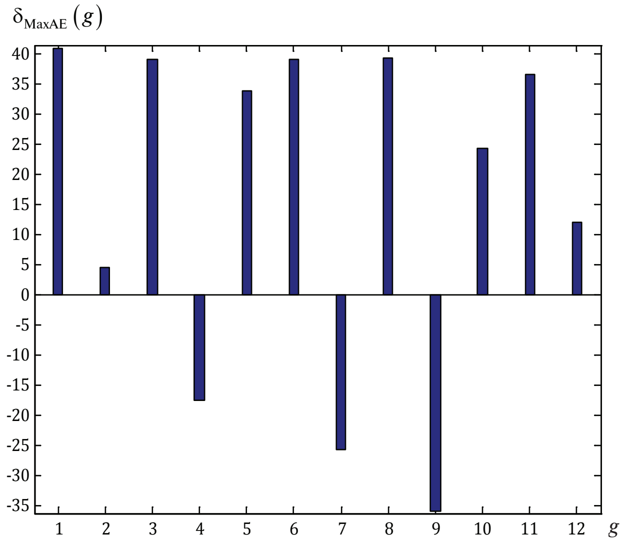


Fig. 7. Difference (46) in the case of a single LSTM layer

For the MaxAE minimization, there are two such instances (see Fig. 6). Differences (39) and (46) are distinctly negative at $g = 9$ (see Fig. 5 and 7), that is an indicator of integers $\{\lambda^{*}_{\mathrm{MaxAE}}(9, z)\}_{z=1}^{50}$ are significantly large, so the difference between the number of the LSTM layer size and consistent lags (in the case of a single LSTM layer) is quite big.

On the contrary, the best forecasting accuracy is achieved by defining a number of LSTM layer neurons as the lag in a time series generated by pattern (8). This is confirmed by Fig. 6, by which roughly a half of the time series instances are forecasted with a minimal MaxAE by the LSTM layer size definition.

For the case of the two LSTM layers network (Fig. 3), RMSE (17) by (13)—(16) for (20)—(23) and (24)—(31) is calculated for each quadruple of g, z, k, m, where k and m are indices of the first and second LSTM layer size. Then, for set

$$\{\rho_{\mathrm{RMSE}}(g, z, k, m)\}_{k=1}^{22} \tag{47}$$

two indices

$$[k^{*(1)}_{\mathrm{RMSE}} \quad m^{*(1)}_{\mathrm{RMSE}}] \in \arg \min_{k=\overline{1,\,22},\; m=\overline{1,\,22}} \rho_{\mathrm{RMSE}}(g, z, k, m) \tag{48}$$

at which RMSEs (47) are minimal are found, wherein $h_{**}(g, z, k^{*(1)}_{\mathrm{RMSE}})$ and $h_{**}(g, z, m^{*(1)}_{\mathrm{RMSE}})$ are the minimal numbers of neurons in the first and second LSTM layers, respectively. As previously, these indices are compared to the consistent lags for the g-th pattern and z-th time series instance. Integer

$$\lambda^{*(1)}_{\mathrm{RMSE}}(g, z) = \min_{l=\overline{1,\,L(g,\,z)}} \left| h_{**}(g, z, k^{*(1)}_{\mathrm{RMSE}}) - p_l(g, z) \right| \tag{49}$$

is the shortest distance (in neurons) between the first LSTM layer size and consistent lags; integer

$$\lambda^{*(2)}_{\mathrm{RMSE}}(g, z) = \min_{l=\overline{1,\,L(g,\,z)}} \left| h_{**}(g, z, m^{*(1)}_{\mathrm{RMSE}}) - p_l(g, z) \right| \tag{50}$$

is the shortest distance (in neurons) between the second LSTM layer size and consistent lags. In addition, distance

$$\lambda^{*}_{\mathrm{RMSE}}(g, z) = \min_{l=\overline{1,\,L(g,\,z)}} \sqrt{\begin{array}{l}(h_{**}(g, z, k^{*(1)}_{\mathrm{RMSE}}) - p_l(g, z))^2 + \\ (h_{**}(g, z, m^{*(1)}_{\mathrm{RMSE}}) - p_l(g, z))^2\end{array}} \tag{51}$$

by (48)—(50) is calculated.

The number of instances when (36) is true is summed up along $z = \overline{1,\,50}$ (Fig. 8). The number of instances when

$$h_{**}(g, z, m^{*(1)}_{\mathrm{RMSE}}) \in P(g, z) \tag{52}$$

is true is summed up along $z = \overline{1,\,50}$ also (Fig. 9). These numbers are summed as well (Fig. 10). Besides, averages

$$\tilde{\lambda}^{*(1)}_{\mathrm{RMSE}}(g) = \frac{1}{50} \cdot \sum_{z=1}^{50} \lambda^{*(1)}_{\mathrm{RMSE}}(g, z) \tag{53}$$

and

$$\tilde{\lambda}^{*(2)}_{\mathrm{RMSE}}(g) = \frac{1}{50} \cdot \sum_{z=1}^{50} \lambda^{*(2)}_{\mathrm{RMSE}}(g, z) \qquad (54)$$

are calculated, and (37) is calculated by (51). Averages (53), (54), (37) are compared to (38). The differences

$$\delta^{(1)}_{\mathrm{RMSE}}(g) = \bar{\lambda}(g) - \tilde{\lambda}^{*(1)}_{\mathrm{RMSE}}(g), \qquad (55)$$

$$\delta^{(2)}_{\mathrm{RMSE}}(g) = \bar{\lambda}(g) - \tilde{\lambda}^{*(2)}_{\mathrm{RMSE}}(g), \qquad (56)$$

$$\delta_{\mathrm{RMSE}}(g) = \bar{\lambda}(g) - \tilde{\lambda}^{*}_{\mathrm{RMSE}}(g) \qquad (57)$$

are shown in Fig. 11.

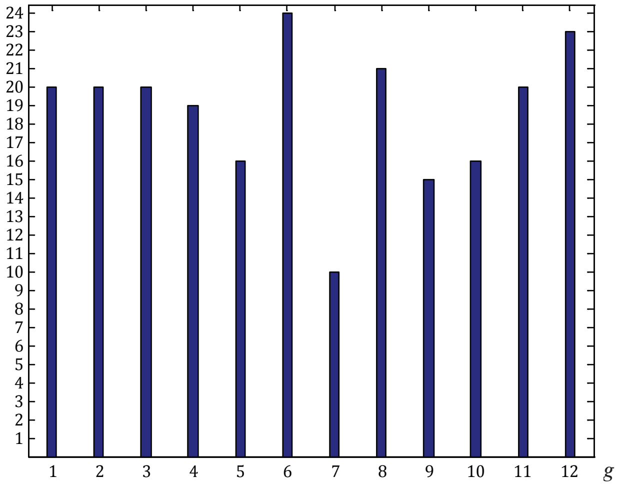


Fig. 8. The number of the LSTM-layer-size and consistent-lag coincidences by (36) in the first LSTM layer (the possible maximum of this number is 50)

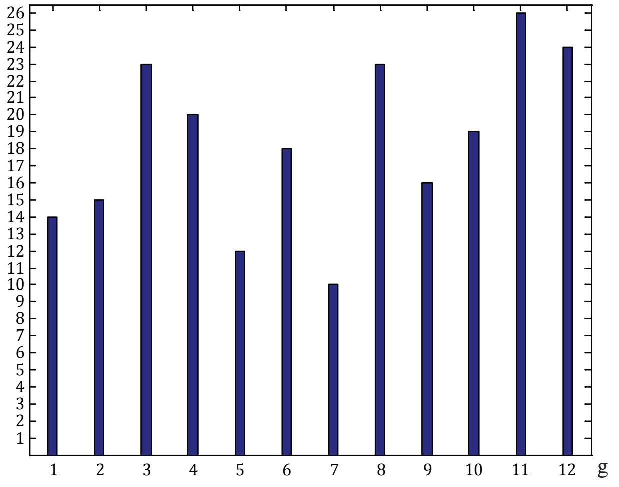


Fig. 9. The number of the LSTM-layer-size and consistent-lag coincidences by (52) in the second LSTM layer (the possible maximum of this number is 50)

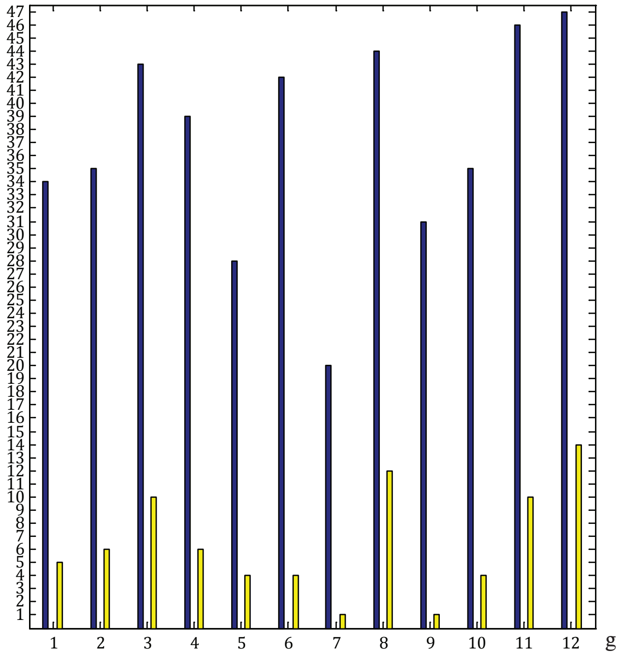


Fig. 10. The sum of instances when either of (36) and (52) is true (the possible maximum of this number is 100), and when both (36) and (52) are simultaneously true (lighter-coloured bars; the possible maximum of this number is 50)

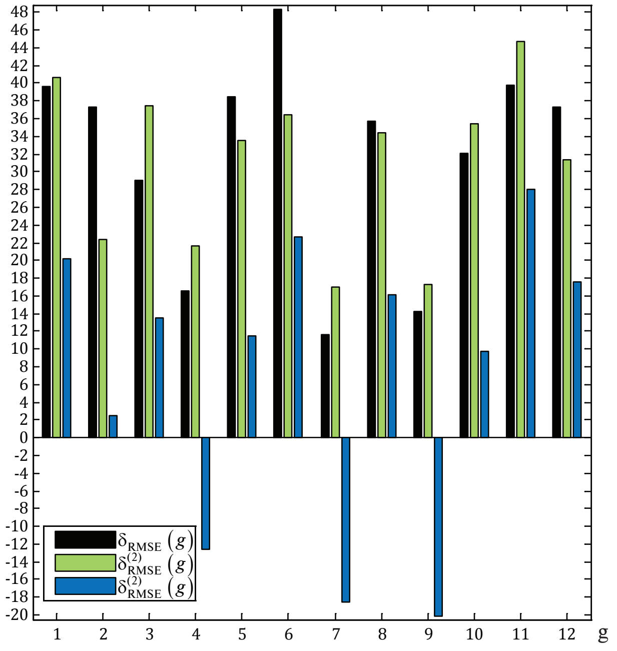


Fig. 11. Polylines of differences (55)—(57)

Figs. 8—11 confirm the inference from Figs. 4—7 about patterns (4), (7), (9): the best forecasting RMSE-accuracy can hardly be achieved by defining a number of LSTM layer neurons (in each layer) as the lag in a time series generated by those patterns. This inference may be weakened for patterns (4) and (9), when only the first LSTM-layer size is set at a consistent lag (because the pattern with a linear trend with exponential extinction is Fig. 8 clearly appears to be the worst). Although the second LSTM-layer-size setting appears to be more chaotic with respect to the lag (Fig. 9), the grand total of the coincidences by (52) is 220 versus 224 coincidences by (36). That is, about a third of minimal-RMSE instances (out of 600) have been obtained by setting the size of one of the LSTM layers at a consistent lag. The grand total of instances, when the sizes of both the layers are set at consistent lags, is just 77 (see lighter-coloured bars in Fig. 10) out of 600.

For each quadruple of g, z, k, m, MaxAE (18) by (13)—(16) for (20)—(23) and (24)—(31) is calculated for the case of the two LSTM layers network (Fig. 3) likewise: for set

$$\{\rho_{\text{MaxAE}}(g, z, k, m)\}_{k=1}^{22} \tag{58}$$

two indices

$$[k^{*(1)}_{\text{MaxAE}} \quad m^{*(1)}_{\text{MaxAE}}] \in \arg \min_{k=\overline{1,\,22},\, m=\overline{1,\,22}} \rho_{\text{MaxAE}}(g, z, k, m) \tag{59}$$

at which MaxAEs (58) are minimal are found, wherein $h_{**}(g, z, k^{*(1)}_{\text{MaxAE}})$ and $h_{**}(g, z, m^{*(1)}_{\text{MaxAE}})$ are the minimal numbers of neurons in the first and second LSTM layers, respectively. As previously, these indices are compared to the consistent lags for the g-th pattern and z-th time series instance: distances

$$\lambda^{*(1)}_{\text{MaxAE}}(g, z) = \min_{l=\overline{1,\, L(g,\, z)}} \left| h_{**}(g, z, k^{*(1)}_{\text{MaxAE}}) - p_l(g, z) \right|, \tag{60}$$

$$\lambda^{*(2)}_{\text{MaxAE}}(g, z) = \min_{l=\overline{1,\, L(g,\, z)}} \left| h_{**}(g, z, m^{*(1)}_{\text{MaxAE}}) - p_l(g, z) \right|, \tag{61}$$

and

$$\lambda^{*}_{\text{MaxAE}}(g, z) = \min_{l=\overline{1,\, L(g,\, z)}} \sqrt{\begin{array}{l}(h_{**}(g, z, k^{*(1)}_{\text{MaxAE}}) - p_l(g, z))^2 + \\ (h_{**}(g, z, m^{*(1)}_{\text{MaxAE}}) - p_l(g, z))^2\end{array}} \tag{62}$$

by (59)—(61) are calculated.

The number of instances when (44) is true is summed up along $z = \overline{1, 50}$ (Fig. 12). The number of instances when

$$h_{**}(g, z, m^{*(1)}_{\text{MaxAE}}) \in P(g, z) \tag{63}$$

is summed up along $z = \overline{1, 50}$ also (Fig. 13). These numbers are summed as well (Fig. 14). Besides, averages

$$\tilde{\lambda}^{*(1)}_{\text{MaxAE}}(g) = \frac{1}{50} \cdot \sum_{z=1}^{50} \lambda^{*(1)}_{\text{MaxAE}}(g, z), \tag{64}$$

and

$$\tilde{\lambda}^{*(2)}_{\text{MaxAE}}(g) = \frac{1}{50} \cdot \sum_{z=1}^{50} \lambda^{*(2)}_{\text{MaxAE}}(g, z) \tag{65}$$

are calculated, and (45) is calculated by (62). Averages (64), (65), (45) are compared to (38). The differences

$$\delta^{(1)}_{\text{MaxAE}}(g) = \bar{\lambda}(g) - \tilde{\lambda}^{*(1)}_{\text{MaxAE}}(g), \tag{66}$$

$$\delta^{(2)}_{\text{MaxAE}}(g) = \bar{\lambda}(g) - \tilde{\lambda}^{*(2)}_{\text{MaxAE}}(g), \tag{67}$$

$$\delta_{\text{MaxAE}}(g) = \bar{\lambda}(g) - \tilde{\lambda}^{*}_{\text{MaxAE}}(g) \tag{68}$$

are shown in Fig. 15.

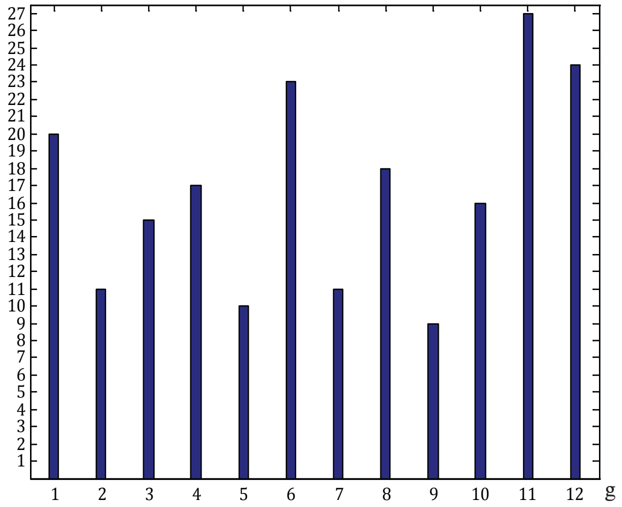


Fig. 12. The number of the LSTM-layer-size and consistent-lag coincidences by (44) in the first LSTM layer (the possible maximum of this number is 50)

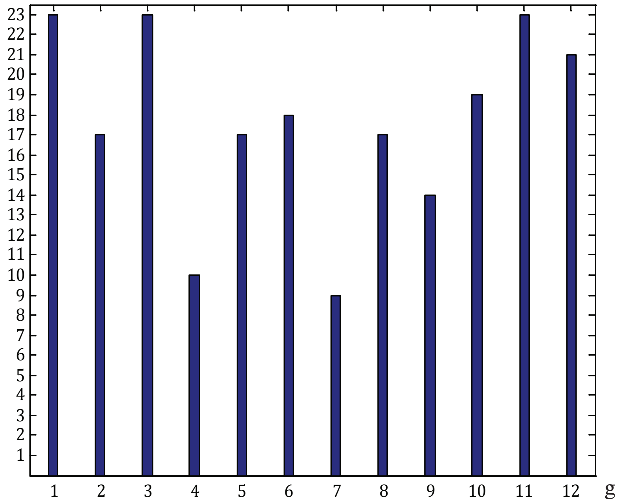


Fig. 13. The number of the LSTM-layer-size and consistent-lag coincidences by (63) in the second LSTM layer (the possible maximum of this number is 50)

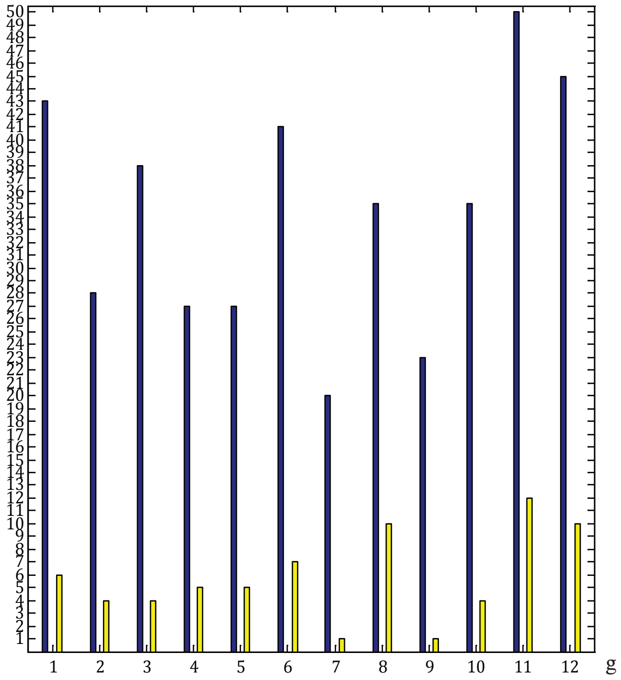


Fig. 14. The sum of instances when either of (44) and (63) is true (the possible maximum of this number is 100), and when both (44) and (63) are simultaneously true (lighter-coloured bars; the possible maximum of this number is 50)

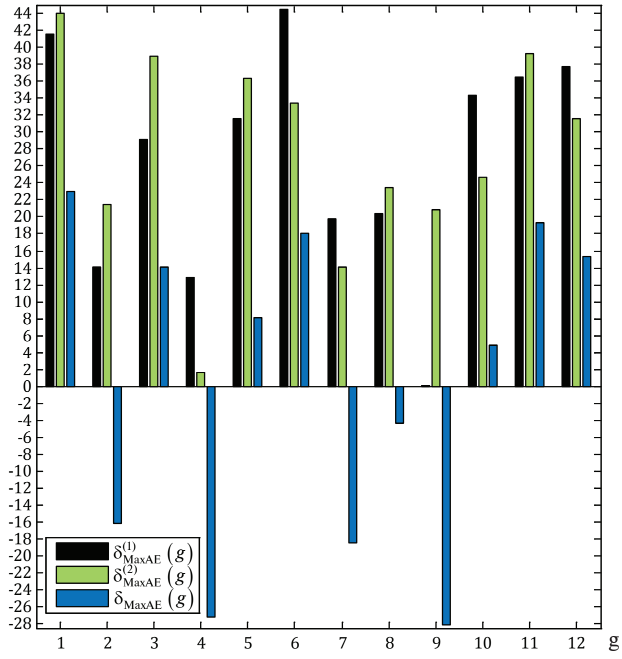


Fig. 15. Polylines of differences (66)–(68)

Fig. 12–15 confirm the inference from Fig. 4–11 about patterns (4), (7), (9): the minimal MaxAE is unachievable by defining a number of LSTM layer neurons (in each layer) as the lag in a time series generated by those patterns. Despite of $\delta_{\mathrm{MaxAE}}(2) < 0$ and $\delta_{\mathrm{MaxAE}}(8) < 0$ (Fig. 15), this inference cannot be supplemented by patterns (2) and (8) because the respective bars in Figs. 12–14 are not that low. The grand total of the coincidences by (44) is 201 versus 211 coincidences by (63). As in the case of RMSE, about a third of minimal-MaxAE instances (out of 600) have been obtained by setting the size of one of the LSTM layers at a consistent lag. The grand total of instances, when the sizes of both the layers are set at consistent lags, is fully comparable to that of RMSE: it is just 69 (see lighter-coloured bars in Fig. 14) out of 600.

In the case of the two LSTM layers network, the best forecasting accuracy is achieved by defining a number of LSTM layer neurons as the lag in a time series generated by patterns (1), (3), (6), (8), (11), (12). This is confirmed by considering Fig. 8–15 in the aggregate, where the hardest-to-handle patterns are (7) and (9). Time series by these patterns (linear trend and exponential extinction with or without seasonality, see the seventh and ninth subplot rows in Fig. 1) are worst-forecasted in the case of a single LSTM layer, which is straightforwardly confirmed by Fig. 4, 5, 7.

Discussion

The abovementioned inferences imply that time series with a linear trend are forecasted worst, whereas the proper LSTM-layer-size-to-lag setting (adjustment) does not help much. The best-forecasted are time series with only repeated random subsequences, or seasonality, or exponential rising. Meanwhile, adding the second LSTM layer improves the performance on average: the two LSTM layers networks have provided 15.6 % decreased RMSE and 18.8 % decreased MaxAE. Adding the third LSTM layer, apart from slowing down the forecasting routine, does not improve the forecasting accuracy.

Fig. 4, 6, 8–10, 12–14 show that over 50 % of the LSTM-layer-size-to-lag adjustment have not provided the minimum of either RMSE or MaxAE. On the contrary, Fig. 5, 7, 11, 15 show that minimal RMSE and MaxAE are obtained by the size set closer to the lag. Consequently, time series lags are distinctive pivots, at which the LSTM layer size is recommended to be set as close as possible.

Unexpectedly, but 46 % of minimal-RMSE instances obtained by the exact LSTM-layer-size-to-lag adjustment correspond to the LSTM network architecture, where the second LSTM layer size is less than the first layer size. It is 47.2 % for minimal-MaxAE instances. In other words, in more than 52 % of best-accuracy instances, the second LSTM layer size is not less than the first layer size.

Conclusions

Based on the computational study results, it is ascertained that the approximately best forecasting accuracy may be expectedly achieved by setting the number of LSTM layer neurons at the time series lag. In probabilistic terms, the best forecasting accuracy likelihood is greater than 30 %. Nevertheless, it is roughly 50 % probable that the exact setting will not provide the minimum (of either RMSE or MaxAE). So, the best forecasting accuracy cannot be guaranteed. However, compared to the single LSTM layer network, it is improved by 15 % to 19 % by applying the two LSTM layers network. Therefore, LSTM networks for time series forecasting can be optimized by using only two LSTM layers whose size is set at the time series lag. Some discrepancy between the size and lag is still acceptable, though. The size of the second LSTM layer should not be less than the size of the first layer.

References

[1] B. Schelter, M. Winterhalder, and J. Timmer, *Handbook of Time Series Analysis: Recent Theoretical Developments and Applications*, Wiley, 2006, doi: 10.1002/9783527609970.

[2] V. Kotu and B. Deshpande, “Chapter 10. Time Series Forecasting”, in: *Predictive Analytics and Data Mining*, Kotu V. and Deshpande B., Eds., Morgan Kaufmann, 2015, pp. 305–327, doi: 10.1016/B978-0-12-801460-8.00010-0.

[3] V. Kotu and B. Deshpande, “Chapter 12. Time Series Forecasting”, in: *Data Science, 2nd ed.*, Kotu V. and Deshpande B., Eds., Morgan Kaufmann, 2019, pp. 395–445, doi: 10.1016/B978-0-12-814761-0.00012-5.

[4] J. G. De Gooijer and R. J. Hyndman, “25 years of time series forecasting”, *Int. J. of Forecasting*, vol. 22, no. 3, pp. 443–473, 2006, doi: 10.1016/j.ijforecast.2006.01.001.

[5] R. DiPietro and G. D. Hager, “Chapter 21. Deep learning: RNNs and LSTM”, in: *Handbook of Medical Image Computing and Computer Assisted Intervention*, Zhou S. K., Rueckert D., and Fichtinger G., Eds., Academic Press, 2020, pp. 503–519, doi: 10.1016/B978-0-12-816176-0.00026-0.

[6] M. Fakhfekh and A. Jeribi, “Volatility dynamics of crypto-currencies’ returns: Evidence from asymmetric and long memory GARCH models”, *Res. in Int. Bus. and Finance*, vol. 51, 101075, 2020, doi: 10.1016/j.ribaf.2019.101075.

[7] M. Sangiorgio and F. Dercole, “Robustness of LSTM neural networks for multi-step forecasting of chaotic time series”, *Chaos, Solitons & Fractals*, vol. 139, 110045, 2020, doi: 10.1016/j.chaos.2020.110045.

[8] V. V. Romanuke, “Regard of parameters and quality of forecast in selecting the neural net optimal architecture for a problem of the time series neuronet forecasting”, *Sci. and Econ.*, no. 3 (27), pp. 164–168, 2012.

[9] G. Box *et al.*, *Time Series Analysis: Forecasting and Control*, Prentice Hall, Englewood Cliffs, NJ, 1994.

[10] V. V. Romanuke, “Decision making criteria hybridization for finding optimal decisions’ subset regarding changes of the decision function”, *J. of Uncertain Syst.*, vol. 12, no. 4, pp. 279–291, 2018.

[11] R. Kneusel, *Random Numbers and Computers*, Springer International Publishing, 2018, doi: 10.1007/978-3-319-77697-2.

[12] V. V. Romanuke, “Time series smoothing and downsampling for improving forecasting accuracy”, *Appl. Comput. Syst.*, vol. 26, no. 1, pp. 60–70, 2021, doi: 10.2478/acss-2021-0008.

[13] R. E. Edwards, *Functional Analysis. Theory and Applications*, Hold, Rinehart and Winston, 1965.

[14] V. V. Romanuke, “Wind speed distribution direct approximation by accumulative statistics of measurements and root-mean-square deviation control”, *Elect., Control and Commun. Eng.*, vol. 16, no. 2, pp. 65–71, 2020, doi: 10.2478/ecce-2020-0010.

[15] F. C. Pereira and S. S. Borysov, “Machine Learning Fundamentals”, in: *Mobility Patterns, Big Data and Transport Analytics*, Antoniou C., Dimitriou L., and Pereira F., Eds., Elsevier, 2019, pp. 9–29. doi: 10.1016/B978-0-12-812970-8.00002-6.

[16] J.-T. Chien, “Deep Neural Network”, in: *Source Separation and Machine Learning*, Chien J.-T., Ed., Academic Press, 2019, pp. 259–320. doi 10.1016/B978-0-12-804566-4.00019-X.

В. В. Романюк

ОПТИМІЗАЦІЯ LSTM-МЕРЕЖ ДЛЯ ПРОГНОЗУВАННЯ ЧАСОВИХ РЯДІВ

Проблематика. Нейронні LSTM-мережі є надзвичайно перспективним засобом для розвитку аналізу та прогнозування часових рядів. Однак, як і нейронні мережі для інших дисциплін та застосувань, LSTM-мережі мають низку версій архітектури, параметрів навчання і гіперпараметрів, неправильний підбір яких може призвести до неприйнятно поганої продуктивності (поганих або дуже ненадійних прогнозів). Тому питання оптимізації LSTM-мереж все ще є відкритим.

Мета дослідження. Встановити, чи досягається найкраща точність прогнозування за такої кількості нейронів у LSTM-шарі, яку можна визначити за лагом часового ряду.

Методика реалізації. Для досягнення поставленої мети пропонується набір контрольних часових рядів для тестування точності прогнозування. Далі визначається порядок обчислювального дослідження для різних версій LSTM-мережі. У підсумку проводиться повна візуалізація й обговорення результатів обчислювального дослідження.

Результати дослідження. Найгірше прогнозуються часові ряди з лінійним трендом, а визначення розміру LSTM-шару за лагом у часовому ряді не дуже допомагає. Найкраще прогнозуються часові ряди, що мають лише повторювані випадкові підпослідовності, або сезонність, або експоненціальне зростання. При застосуванні мережі з двома LSTM-шарами точність прогнозування покращується на 15...19 % як порівняти з мережею з одним LSTM-шаром.

Висновки. Приблизно найкраща точність прогнозування може бути очікувано досягнута за встановлення числа нейронів у LSTM-шарі, яке дорівнює лагу часового ряду. Однак це не гарантує найкращу точність прогнозування. LSTM-мережі для прогнозування часових рядів можуть бути оптимізовані за використання лише двох LSTM-шарів, розмір яких дорівнює лагу часового ряду. Втім, деякі розбіжності є також прийнятними. Розмір другого LSTM-шару має бути не меншим від розміру першого.

Ключові слова: прогнозування часових рядів; LSTM-мережа; розмір LSTM-шару; точність прогнозування; середньоквадратична помилка; максимальна абсолютна похибка.

В. В. Романюк

ОПТИМИЗАЦИЯ LSTM-СЕТЕЙ ДЛЯ ПРОГНОЗИРОВАНИЯ ВРЕМЕННЫХ РЯДОВ

Проблематика. Нейронные LSTM-сети являются чрезвычайно перспективным средством для развития анализа и прогнозирования временных рядов. Однако, как и нейронные сети для других дисциплин и приложений, LSTM-сети имеют множество версий архитектуры, параметров обучения и гиперпараметров, неадекватный подбор которых может привести к неприемлемо плохой производительности (плохим или очень ненадёжным прогнозам). Поэтому вопрос оптимизации LSTM-сетей всё ещё остаётся открытым.

Цель исследования. Установить, достигается ли наилучшая точность прогнозирования при таком количестве нейронов в LSTM-слое, которое может быть определено по лагу временного ряда.

Методика реализации. Для достижения поставленной цели предлагается набор контрольных временных рядов для тестирования точности прогнозирования. Далее определяется порядок вычислительного исследования для различных версий LSTM-сетей. В конечном итоге, проводится полная визуализация и обсуждение результатов вычислительного исследования.

Результаты исследования. Хуже всего прогнозируются временные ряды с линейным трендом, а определение размера LSTM-слоя по лагу во временном ряде не особо помогает. Лучше всего прогнозируются временные ряды, имеющие только повторяемые случайные подпоследовательности, или сезонность, или экспоненциальный рост. В случае применения сети с двумя LSTM-слоями по сравнению с сетями с одним LSTM-слоем точность прогнозирования улучшается на 15...19 %.

Выводы. Приблизительно наилучшая точность прогнозирования может быть ожидаемо достигнута при установлении числа нейронов в LSTM-слое равным лагу временного ряда. Однако это не гарантирует наилучшей точности прогнозирования. LSTM-сети для прогнозирования временных рядов могут быть оптимизированы при использовании только двух LSTM-слоёв, чей размер устанавливается равным лагу временного ряда. Впрочем, некоторые расхождения также приемлемы. Размер второго LSTM-слоя должен быть не меньше размера первого.

Ключевые слова: прогнозирование временных рядов; LSTM-сеть; размер LSTM-слоя; точность прогнозирования; среднеквадратическая ошибка; максимальная абсолютная погрешность.

Рекомендована Радою
факультету прикладної математики
КПІ ім. Ігоря Сікорського

Надійшла до редакції
9 травня 2021 року

Прийнята до публікації
7 грудня 2021 року