Journal homepage: https://itce.vn.ua/en

Information Technologies and Computer Engineering **Vol. 23, No. 1. 2026**

Article's History: Received: 27.10.2025 Revised: 23.01.2026
Accepted: 26.03.2026 Published: 20.04.2026

ISSN 1999-9941; e-ISSN 2078-6387
UDC 004.85
DOI: 10.31649/vitce/1.2026.72

Forecasting of time series using a neural network with parallel-stacked LSTM blocks

Yurii Futryk*
Postgraduate Student
Lviv Polytechnic National University
79000, 12 Stepan Bandera Str., Lviv, Ukraine
https://orcid.org/0000-0001-5271-9883

Ivan Peleshchak
PhD, Associate Professor
Lviv Polytechnic National University
79000, 12 Stepan Bandera Str., Lviv, Ukraine
https://orcid.org/0000-0002-7481-8628

Abstract. Time series forecasting is crucial for supporting decisions in financial analytics, where data is characterised by non-linearity, non-stationarity, and high noise levels. The purpose of the study was to determine the effective configuration of a recurrent neural network with a parallel combination of Long Short-Term Memory (LSTM) cell stacks to improve the accuracy of stock price forecasting, and the possibilities of applying the back-end model in industry, energy, and related domains. The study applied deep learning methods using the TensorFlow/Keras library, and used historical data from Google shares to train the model. It was established that the architecture with parallel-stacked blocks provided higher learning stability compared to standard recurrent models due to more efficient allocation of technical features of the time sequence. It has been experimentally proven that the optimal number of neurons in the hidden layers for such a task was 100-200 units, while a further increase in the power of the model lead to a retraining effect. It was found that the use of dropout regularisation in the range of 0.1-0.2 minimised the error in the validation sample, while values over 0.3 significantly slowed down the convergence of the algorithm. Feature analysis showed that integrating an exponential moving average with a short time window improved the model result, showing a higher correlation with the target index than the relative strength index. The prediction quality of the model was evaluated by the Mean Squared error (MSE), the Root Mean Squared Error (RMSE), and the Mean Absolute Percentage Error (MAPE). It was found that configurations (50-100 blocks) were characterised by increased MAPE values, while in the range of 180-400 blocks the error decreased and became stable. The most accurate result was obtained for a configuration with 325 blocks, Dropout regularisation = 0.05 and Nadam optimiser (Nesterov-accelerated Adam): MAPE = 1.62%, RMSE = 2.41, MSE = 6.05. The practical significance of the study lied in the formulation of clear recommendations for setting up hyperparameters of LSTM models for applied short-term forecasting of financial series

Keywords: deep learning; Dropout-regularisation; Nadam-optimisation; EMA; RSI

Introduction

The task of predicting time series is one of the most important challenges of applied analytics in the 2010-2026 – it appears in various domains from exchange analysis and network load to high-frequency sensor data, where long and short inter-time dependencies are manifested. Reproducibility of evaluation procedures (time split, stable normalisation, non-mixing of samples), and transparency of model settings remain a priority for researchers and practitioners.

In the course of forecasting financial time series, there is a need for methods that combine practical efficiency with ensuring transparency and reproducibility of the results obtained. Long Short-Term Memory (LSTM) neural networks are widely used in many industries to predict time series, such as detecting equipment failures, predicting production line loads, and improving logistics management efficiency. Due to its ability to model long-term dependencies, LSTMs

Suggested Citation:
Futryk, Yu., & Peleshchak, I. (2026). Forecasting of time series using a neural network with parallel-stacked LSTM blocks. *Information Technologies and Computer Engineering*, 23(1), 72-82. doi: 10.31649/vitce/1.2026.72

*Corresponding author

allow achieving high accuracy in complex forecasts, especially in finance and other critical areas.

In the global scientific literature on financial forecasting, the problem of practical selection of the configuration of LSTM models (number of blocks, strength of regularisation, and composition of features) remains, since accuracy indicators are sensitive to training settings and assessment conditions; therefore, the development of reproducible applied recommendations for short-term forecasting is relevant. In contemporary studies of short-term forecasting of financial time series, key attention was paid to the configuration of recurrent LSTM models, the level of Dropout regularisation, and the influence of technical indicators EMA (Exponential Moving Average) and RSI (Relative Strength Index) on the accuracy of forecasts. For example, O.B. Sezer *et al*. (2020) conducted a review of deep learning in financial forecasting and showed that incorrect time separation and different preprocessing often distort model comparisons. The researchers emphasised the need for a reproducible protocol (chronological division into training and test data, uniform scaling) and transparent reporting of MAPE (Mean Absolute Percentage Error) and RMSE (Root Mean Squared Error) metrics.

I.E. Livieris *et al*. (2020) investigated hybrid architectures that combine CNN (convolutional neural network) and LSTM (recurrent neural network). In such architectures, CNNs are used to isolate local patterns in data, and LSTMs are used to model time dependencies. Researchers have shown that a more complex architecture can improve the quality of prediction, but simultaneously make the model more sensitive to hyperparameter settings (for example, the number of layers or regularisation level). This means that for correct comparison of such models, it is necessary to provide controlled conditions with the same settings for all models. In turn, P.T. Yamak *et al*. (2020) compared statistical and neural network methods for predicting financial time series and noted that the advantages of recurrent models, such as LSTM, are manifested if the observation window and model settings are correctly selected. They stressed that the correctness of conclusions about the effectiveness of models depends on the same conditions for preparing data and evaluation metrics. This confirmed the need to use the same protocols for correct comparison of models.

In a similar context, S. Smyl (2020) demonstrated the effectiveness of combining statistical methods, such as exponential smoothing, with neural networks for predicting time series. The key conclusion of their study was that the values of forecast errors make sense only if the forecast horizon and evaluation protocol were clearly defined. If these conditions were not considered, the advantage of one model over another may be the result of different evaluation conditions or data, rather than architecture. B. Lindemann *et al*. (2021) considered another important issue – the reproducibility of results when using LSTM to predict time series. They found that the stability of the results largely depends on factors such as model capacity (hidden state parameters and number of cells), regularisation, and validation procedure. The researchers noted that recommendations for choosing hyperparameters (for example, the number of blocks or regularisation parameters) can only be valid if there is a complete description of the training and testing protocol. This highlights the importance of clear and transparent reporting of model settings, which allows achieving reproducibility of results and comparing models under the same conditions.

In this context, B. Lim *et al*. (2021) drew attention to the importance of controlled experiments when comparing optimisation algorithms. They stressed that the same data preparation, regularisation, forecast horizons, and evaluation metrics should be used to correctly compare optimisers. If these conditions are not met, conclusions about the superiority of one optimiser over another may be unstable, since the result will depend on changes in settings, and not on the quality of the optimiser itself. Additionally, M. Ez-zaiym *et al*. (2025) gave an example of a controlled comparison of Adam (Adaptive Moment Estimation) and Nadam (Nesterov-accelerated Adaptive Moment Estimation) optimisers under agreed training conditions. The researchers showed that generalising the advantage of one optimiser over another is limited without fixing architectural parameters (such as model capacity and Dropout level) and other settings. Therefore, the optimiser should be interpreted as part of a holistic model configuration, where all components must be configured in a single context to achieve reliable results.

According to H. Widiputra *et al*. (2021), changes in the composition of input features in the multivariate formulation of financial forecasting can significantly affect quality metrics, even if the model architecture remains unchanged. They stressed that the contribution of technical indicators should be evaluated only under fixed preprocessing and the same normalisation conditions. This approach avoids mixed effects that can distort the results of model comparisons. H. Abbasimehr & R. Paki (2022) proposed combining LSTM with attention mechanisms for predicting time series, showing that attention helps the model focus on the most relevant parts of history. However, the researchers noted that the gain in accuracy depends on the specific task and does not eliminate the need for systematic selection of hyperparameters and retraining control. Similarly, B. Ghojogh & A. Ghodsi (2023) summarised current approaches to sequence modelling, emphasising the role of "memory" and regularisation mechanisms in the learning stability of recurrent models. They stressed that the competitive quality of models depends not only on the choice of the recurrent block type, but also on the consistency of the evaluation protocol and the correct configuration of hyperparameters for a specific data set.

For the most part, the publications analysed focused on the use of LSTM for predicting financial time series, with an emphasis on practical accuracy. In particular, I. Peleshchak & Y. Futryk (2025) proposed a new neural network configuration with parallel-stacked LSTM blocks, which

significantly improved the accuracy of predictions. They demonstrated that combining LSTM with technical indicators (EMA, RSI) can significantly reduce forecasting errors. Their study highlighted the importance of a systematic approach to setting up models and selecting hyperparameters to achieve stability and accuracy of results.

Furthermore, the analysis of contemporary sources shows that for short-term financial forecasting, applied recommendations for joint adjustment of the number of LSTM blocks/model capacity, the level of Dropout regularisation, and the feasibility of including technical indicators EMA and RSI under the reproducible assessment protocol are not sufficiently systematised. That is why the purpose of the study was to predict and analyse time series with high accuracy on the MSE, RMSE, and MAPE metrics (⩽ 1.9%), and experimental verification of the configuration of the neural network model with parallel-stacked LSTM blocks, considering the exponential mean and relative strength index indicators on the time data set. To achieve this goal, the following tasks were set: to determine the rational configuration of the model by systematically varying the number of LSTM blocks and the level of Dropout regularisation; to assess the contribution of technical indicators EMA and RSI to the quality of forecasting using a fixed data preparation protocol; to conduct a comprehensive assessment of the quality of forecasting using agreed metrics.

Materials and Methods

Experimental database and data selection. A dataset from the well-known company Google, sourced from the open-access platform Yahoo Finance (n.d.), was used to conduct the computer experiment. As part of the study, a dataset was generated based on Google's historical stock prices. The generated dataset covered the period from January 2011 to August 2025 and contained 3,687 records (trading days from the yfinance source). The closing price was chosen as the target variable for forecasting, since this indicator reflected the final valuation of the asset for the trading session and was representative of the analysis of the dynamics of the financial time series.

The experimental data was downloaded from the Python library yfinance (Kurniawan *et al.*, 2024), which provided convenient access to financial indicators. The model was implemented in the Python programming language using the following libraries: NumPy/Pandas (preparation and numerical calculations), Keras (modelling), Matplotlib (visualisation), Yfinance (library for obtaining historical financial data from the Yahoo Finance API source). Technical indicators of the EMA and RSI were additionally calculated to form signs. Flowcharts are visualised using Draw.io (n.d.)

Neural network architecture with parallel-stackable LSTM blocks. The proposed model was a neural network with parallel-stacked LSTM blocks. Figure 1 shows the architecture of a recurrent neural network consisting of parallel LSTM blocks. Standard LSTM cells with input, output, and "forget"-gates were used. Each block in this network processes data received at a specific time point, and through the interaction mechanism between blocks accumulates information from previous time intervals. This principle allowed the model not only to generate output values, but also to adjust its internal state, which ensured higher accuracy.

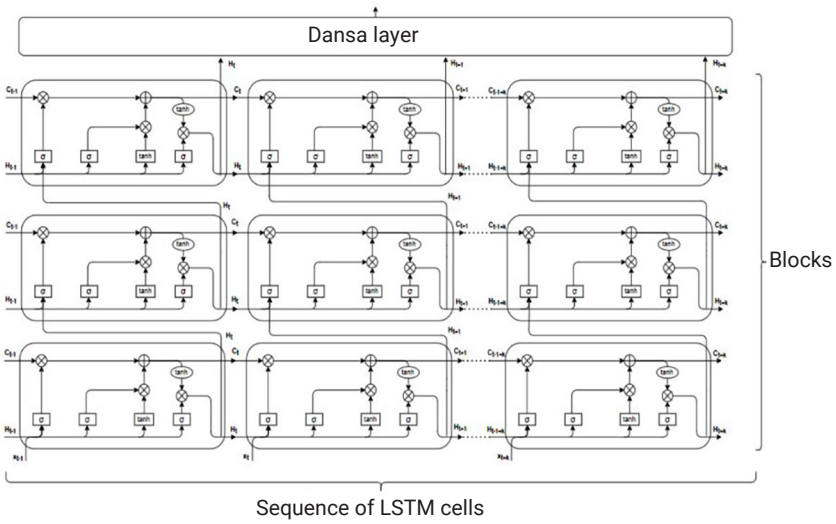

Figure 1. Morphology of a neural network with parallel-stacked LSTM blocks

Source: developed by the authors based on V. Lytvyn *et al*. (2025)

Intermediate views generated by consecutive LSTM layers were then fed to the output tightly coupled layer (Dense), which generates the final forecast value. Within the LSTM blocks themselves (Fig. 2), sigmoid and/or hyperbolic tangent (tanh) nonlinearities are used to construct the hidden state, controlling the activation of the memory cells, and the output signal. The final Dense-layer, which converts these features into scalar prediction, works with linear activation, or can use ReLU in cases where the model must be limited to non-negative or scalable predicted values.

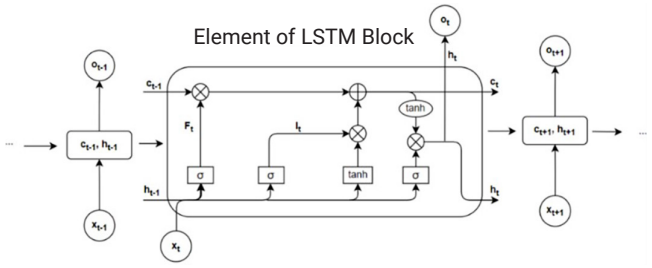

Figure 2. Architecture of a separate LSTM block element
Source: developed by the authors based on Q. Wang & Y. Zhang (2022)

LSTM manages memory sequentially at each step through three independent "solution nodes" (Widiputra *et al.*, 2021): (1) forget something from the previous state; (2) add new information; (3) output a useful segment. The above three-component scheme provided a controlled update of the internal state and reduced the risk of error accumulation when processing long sequences. Further, h_{t-1} indicates the previous hidden state, x_t – current input, and C_{t-1} – state of the memory (cell) in the previous step.

At the first stage, the "forget gate" node is considered – the network viewed the previous hidden state h_{t-1} together with the current input x_t and decide that from an old memory C_{t-1} to leave it. The solution is set by coefficients from 0 to 1 for each memory cell: 0 – erase, 1 – save. Thus, forget gate acts as a selective filter that controls the share of stored information in memory.

$$f_t = \sigma(W_t * [h_{t-1}, x_t] + b_f), \quad (1)$$

where f_t – vector of values in the range [0, 1], which determines what proportion of information in the cell C_{t-1} must be saved (1) or forgotten (0); W_t, b_f – weights and offsets that are updated during training; σ – sigmoid activation function.

The next node is "input" (input gate and candidates). Here, the LSTM determines what exactly to add to memory: (a) through the "tolerance node", the network selects which cells are allowed to be updated; (B) separately calculates candidate values bounded by the segment [-1,1]. As a result, the previous memory (after the "forget" node) is added to the selected part of candidates and gives updated memory, creating a vector of new values that are candidates for updating elements. In this way, input gate coordinates "what to update" (tolerance mask) and "what to update" (candidate values), providing managed input of new information.

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i), \quad (2)$$

$$C_t = tan\,h(W_C * [h_{t-1}, x_t] + b_C), \quad (3)$$

where i_t – vector of "activations" that determines the update of information; C_t – vector of candidate values for updating memory.

The final output gate node is responsible for generating a filtered version of the updated memory: first, the network decides which part of the internal state should be "publicised" at the current stage, and then converts it to a new hidden state h_t. It restricts the transfer of secondary memory components and skips only those features that are relevant to the forecast at a given time step. This reduces the risk of random fluctuations (noise) and maintains the stability of hidden state dynamics in long sequences. The resulting hidden state is passed further along the sequence and used for further prediction, in particular, it is fed to the next LSTM block or to the original dense layer in regression problems:

$$o_t = \sigma(W_0 * [h_{t-1}), x_t] + b_0), \quad (4)$$

$$h_t = o_t * tan\,h(C_t), \quad (5)$$

where o_t – vector of output signal values; h_t – updated hidden state passed to the next time step.

In this setting, a neural network with parallel-stacked LSTM blocks implements controlled memory: some scales learn to forget too much, others learn to dose new signals, and others learn to responsibly open the "exit valve". This ensured more stable preservation of important patterns in the time series without noise accumulation. As a result of this principle, a hidden state was formed h_t, which summarised the context of previous steps and was used as input for the next prediction step.

Metrics for evaluating time series prediction by a neural network with parallel-stacked LSTM blocks. Standard MSE, RMSE, and MAPE metrics were used to quantify the quality of the model's prediction, and to identify signs of overtraining, which are described in detail in the paper by D. Chicco *et al.* (2021). MSE was used as a baseline metric to optimise and compare forecasts with actual values in the test set:

$$MSE = \frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2, \quad (6)$$

where y_i – actual value of the target variable for the i-th observation; $\hat{y}_i$ – projected model value; n – number of observations in the sample; i – observation index, $i = 1...n$.

To interpret the error in the units of measurement of the studied variable, RMSE was used, which was calculated from forecasts and actual values in the test sample:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2}, \quad (7)$$

where y_i – actual value of the target variable for the i-th observation; $\hat{y}_i$ – projected model value; n – number of observations in the sample; i – observation index, $i = 1...n$.

Additionally, MAPE was used to represent the percentage error and compare the results between different time intervals/model settings:

$$MAPE = \frac{1}{n}\sum_{i=0}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| * 100\%, \quad (8)$$

where y_i – actual value of the target variable for the i-th observation, $y_i \neq 0$; $\hat{y}_i$ – projected model value; n – number of observations in the sample; $|\cdot|$ – module (absolute value); i – observation index, $i = 1...n$.

Data for the experiment were divided chronologically into train/validation/test samples (without mixing) to avoid information leakage from the future to the past. When comparing models, the target variable Close, input window length 60, Min-Max normalisation (parameters were calculated on "train" and applied to validation/test), and MSE, RMSE, and MAPE metrics were recorded. The number of parallel-stacked LSTM blocks, Dropout, and feature set (with/without EMA and RSI) varied. Results were obtained based on metrics in the test sample, and the difference between train and validation was used to control retraining. The limitation of the experiment was execution for one asset (GOOGL) and one source (Yahoo Finance/yfinance), so generalisation to other instruments and market regimes requires additional verification. Estimates may also vary depending on the input window selection, separation scheme, and hyperparameters.

Results and Discussion

This section presented the results of an experimental test of the performance and accuracy of a neural network with parallel-stacked LSTM blocks on the financial time series of Google shares using a fixed training protocol and the same preprocessing pipeline. To test the stability and impact of architectural solutions, two model configurations were considered, which differed in the number of parallel-stacked LSTM blocks and the Dropout value, while the remaining components of the experiment remained unchanged. Training in both cases was carried out under the same optimisation conditions using the adaptive optimiser Nadam. Figure 3 shows a flowchart of the parallel-stacked LSTM model used in the experiments. The scheme summarises the processing sequence: generating an input window and features, LSTM layers with Dropout, and an output Dense layer that generates a forecast for the target variable Close.

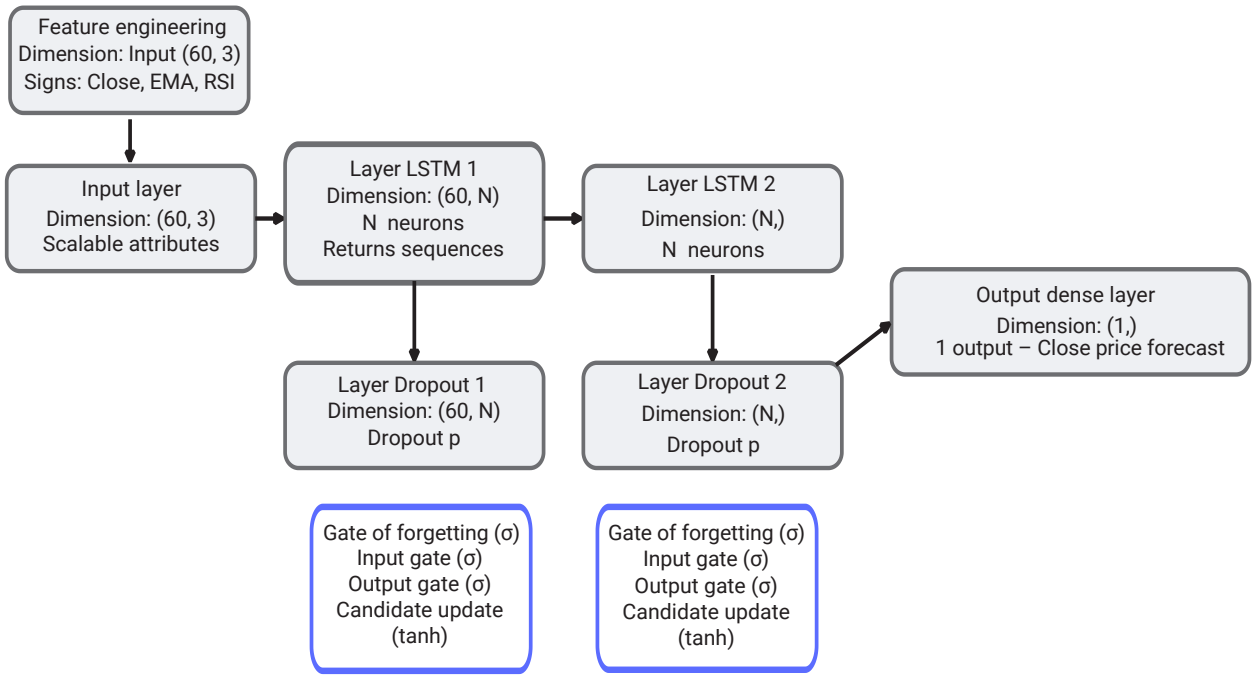

Figure 3. Flowchart of a parallel-stacked LSTM model

Note: N – number of LSTM units in the LSTM layer; p – regularisation coefficient

Source: developed by the authors using the Draw.io tool (n.d.)

As part of the hyperparameter selection, experimental combinations were compared that varied the number of parallel-stacked LSTM blocks and the dropout regularisation level, while the optimiser (Nadam) and feature set (Close, EMA_20, RSI) remained constant. For the final comparison, two configurations were selected: Set A (275; Dropout = 0.10) and Set B (325; Dropout = 0.05), since these settings provided an optimal ratio of accuracy and stability in validation/test using the MSE, RMSE, and MAPE metrics in the fixed training protocol. The total values of the selected hyperparameters and the composition of features for each configuration are shown in Table 1.

Table 1. Values of neural network hyperparameters with parallel-stacked LSTM blocks

Hyperparameters	Number of blocks	Dropout	Optimiser	Technical features
Set A	275	0.1	Nadam	Close, EMA_20, RSI
Set B	325	0.05	Nadam	Close, EMA_20, RSI

Source: developed by the authors based on the results of experiments

Figure 4 shows a comparison of the actual close price dynamics and model predictions for two configurations (A and B) using the same preprocessing and training protocol. A visual assessment was made of how well the forecast matched the actual values on the test segment, and the nature of the errors during periods of sharp trend changes (peaks/dips), where models typically produce the greatest deviations. Special attention was drawn to the gap between the actual series and the forecast at the end of the test interval, since it is most indicative of the model's ability to generalise historical fluctuations. A comparison of subgraphs (A) and (B) demonstrates how changes in the number of LSTM blocks and Dropout affect the stability of the forecast trajectory and noise sensitivity.

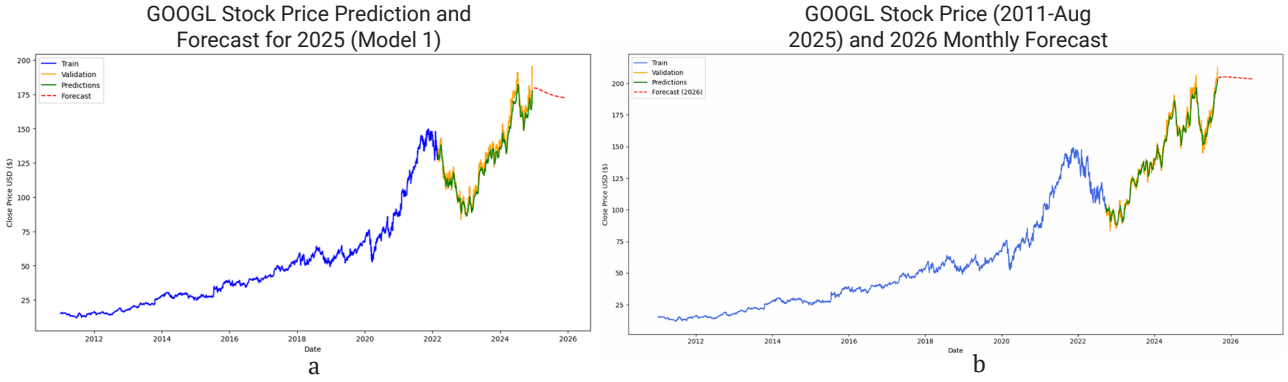

Figure 4. Graphs of stock price forecasts using a neural network with parallel-stacked LSTM blocks

Note: a – for hyperparameters of type A; b – for hyperparameters of type B. Target variable Close, period 2011-2025; A/B configurations according to Table 1. Training, validation, and predicted values are marked with a standard palette, where the green line is the predicted values during verification

Source: compiled by the authors

Figures 5 and 6 illustrate the MAPE dependence on the number of parallel-stacked LSTM blocks (under fixed experimental conditions), which reflects the relationship between the lack of complexity of the model and the risk of retraining. The minimum of the curve corresponds to the area of best alignment of forecasts with actual values in validation/test, so this graph is used to justify the choice of configurations A and B. Increasing the number of blocks does not guarantee improvement, and after a certain limit, the error may increase due to excessive complexity and noise sensitivity. Final configuration comparisons were performed using metrics in the test sample, while validation was used to control retraining.

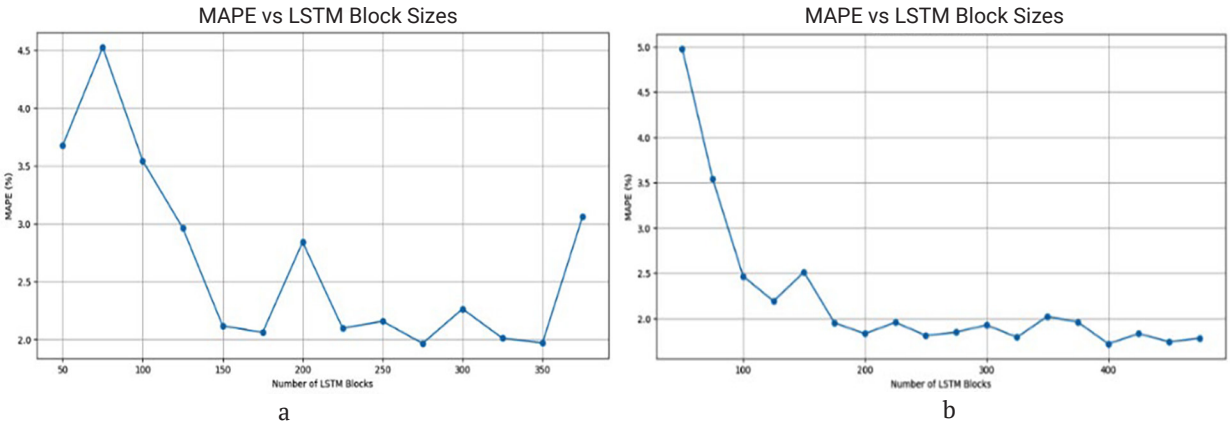

Figure 5. Graph of MAPE metrics that depend on the number of LSTM blocks

Note: a – for hyperparameters of type A; b – for hyperparameters of type B. A/B configurations according to Table 1

Source: compiled by the authors

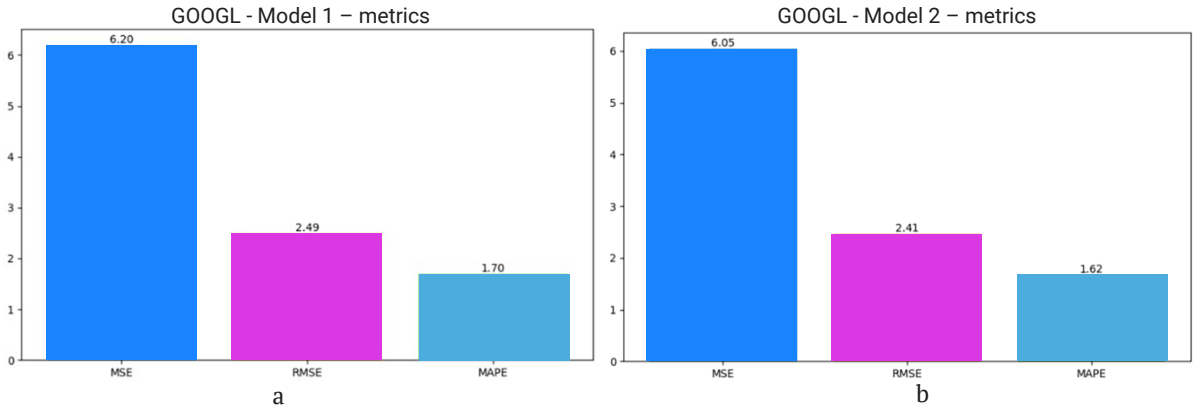

Figure 6. Visualisation of MSE, RMSE, and MAPE metrics based on histograms
Note: a – for 275 LSTM blocks; b – for 325 LSTM blocks
Source: compiled by the authors

Based on the results of a series of experiments, the smallest error on the test sample within the protocol under consideration was obtained for configuration B (N = 325 and Dropout = 0.05), in particular, the MAPE value = 1.62%. Instead, for smaller n values (50-100), MAPE growth and learning instability were observed, and for excessively large Dropout values (⩾ 0.30), convergence deterioration and signs of under-learning were observed. Within the protocol under consideration, the balance between accuracy and generalisation is determined by matching the model capacity (N) and regularisation intensity (Dropout).

In the context of existing approaches in the literature, it is also advisable to consider GRU (Gated Recurrent Unit) – a recurrent neural network with gate mechanisms, which is a more compact alternative to LSTM and usually has fewer parameters. In particular, H. Abbasimehr & R. Paki (2022) proposed a hybrid approach that combines LSTM and multi-head attention, which allowed the model to focus on the most informative fragments of history and better reproduce nonlinear dependencies in time series. A separate area was also financial hybrids for high volatility, where LSTM was combined with statistical models (Koo & Kim, 2022). Simultaneously, this paper focused on a controlled assessment of the proposed architecture, which allowed interpreting the difference in forecast quality as a consequence of changes in capacity and regularisation under constant pipeline conditions.

As noted by Q. Wang & Y. Zhang (2022), as parameterisation of recurrent models increases, the risk of overtraining increases, especially for financial series with noise and structural shifts. In this study, the impact of these factors was evaluated by systematically varying the number of blocks and Dropout with control of validation losses (Fig. 7). Dropout was used as a regularisation mechanism between recurrent components and subsequent layers, which is consistent with typical LSTM regularisation practices.

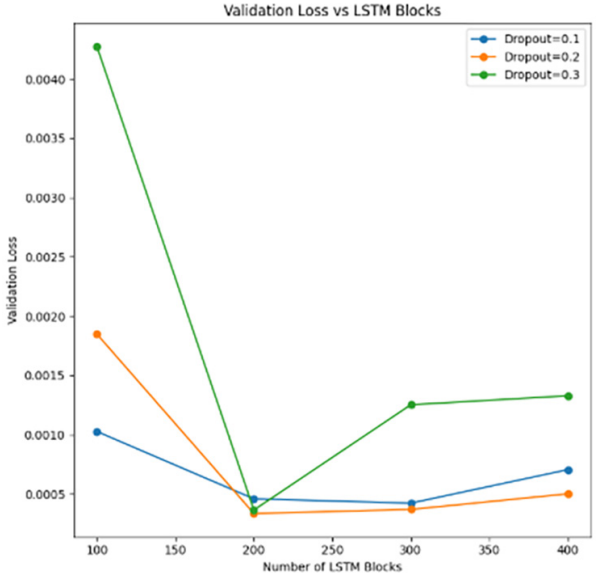

Figure 7. Graph of the dependence of validation losses on the number of LSTM blocks (100-400)
Source: compiled by the authors

The graph shows that excessive regularisation (Dropout = 0.3) leads to significantly higher validation losses at low capacity (100 blocks) and does not provide advantages on larger configurations, while moderate Dropout levels (0.1-0.2) provide lower losses and more stable behaviour when the number of blocks increases. In particular, the lowest values of validation losses are observed around 200 blocks for all Dropout levels considered, and when moving to 300-400 blocks, Dropout = 0.2 shows better stability, which is consistent with the assumption that stronger regularisation is required with increasing model capacity. The results also correlate with the findings of H. Abbasimehr & R. Paki (2022), who emphasised the importance of selecting hyperparameters and regularisation for LSTM approaches in forecasting tasks: although direct numerical comparison is limited by differences in datasets and metrics, the overall trend coincides – correctly selected regularisation and the number of blocks reduce retraining and increase forecast stability.

Figure 8 shows the dependence of the training time on the number of LSTM blocks in the same range (100-400) for different Dropout values. There is an almost quasi-linear increase in the duration of training with an increase in the number of blocks. Accordingly, increasing the capacity of the model increases computational costs due to an increase in the number of parameters and recurrent operations, while the effect of Dropout on training time is secondary (curves for different levels of regularisation are located close to each other). This reflects the computational cost of improving accuracy and is important in a comparative context: lighter statistical or compact neural network models can be faster, but often inferior in reproducing nonlinear dynamics, while LSTM hybrids with attention mechanisms (Abbasimehr & Paki, 2022) or optimised deep LSTM approaches (Gülmez, 2023) can improve accuracy at the cost of increasing learning complexity and parameter matching requirements.

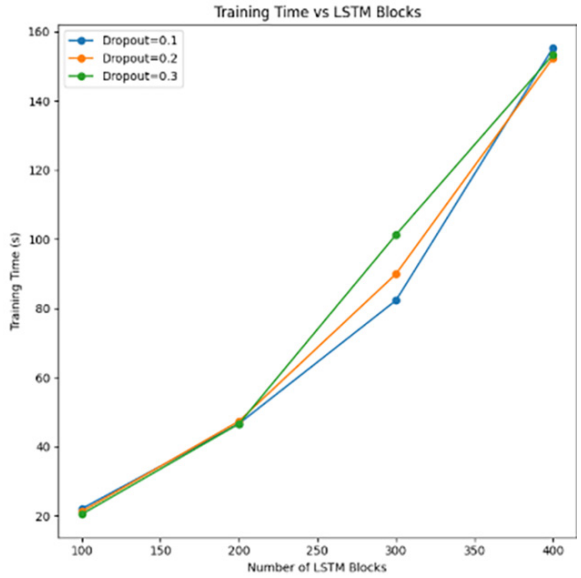

Figure 8. Graph of the curve of dependence of training time on the number of LSTM blocks (100-400)
Source: compiled by the authors

In addition, stock forecasting applications often show that optimisation of LSTM hyperparameters (size/depth, regularisation parameters, optimiser selection) can provide improvements in relation to basic LSTM configurations and statistical models. For example, the study by B. Gülmez (2023) proposed an optimised version of the deep LSTM model for predicting stock prices using a stochastic hyperparameter selection procedure. In parallel with recurrent approaches, architectures based on attention mechanisms are actively developing: Informer and Autoformer have proposed effective solutions for long-term sequencing and scalable forecasting (Wu *et al.*, 2021; Zhou *et al.*, 2021), and described hybrid LSTM-Transformer approaches for financial Series (Kabir *et al.*, 2025), which demonstrated increased reliability and efficiency over longer forecast horizons. Overall, these studies have shown that systematic hyperparameter selection and regularisation are crucial for stability and accuracy. In this context, the current study complemented the existing results by evaluating the contribution of two controlled parameters (N and Dropout) in isolation. The use of EMA and RSI indicators was considered as a compact representation of trend and momentum, but the effect of indicators depends on the forecast horizon, normalisation, and asset properties. Therefore, in this study, a set of features was recorded to separate the influence of architecture from the influence of the feature space. Such fixation was a necessary condition for the correct comparison of architectural configurations.

Conclusions

The research successfully achieved its objective of developing and analysing a method for high-precision

forecasting of financial time series using architecture of parallel-stacked LSTM blocks. Experimental verification of these shares of Google Corporation confirmed the effectiveness of the proposed approach, helping to achieve a forecast error for the MAPE metric at the level of 1.62%, which corresponds to the quality criterion defined in the goal (MAPE ⩽ 1.9%). As part of the study, a representative set of historical data was formed and a pipeline preparation was implemented, which minimises the risk of information leakage between samples due to chronological separation. The proposed neural network architecture, built on several parallel branches with stacking recurrent LSTM layers and then aggregating their outputs, allowed the model to simultaneously process price indicators and technical indicators of the EMA and RSI. Systematic variation of hyperparameters has shown that the optimal balance between computing power and generalisation capacity within a fixed protocol is a configuration with 325 LSTM blocks and a Dropout of 0.05. The use of the Nadam optimiser ensured stable learning convergence, and the obtained MSE and RMSE values consistently confirmed an improvement in the forecast quality for the selected configuration relative to the alternative setting.

The paper conceptualised the advantages of parallel organisation of recurrent structures for analysing non-stationary financial time series. This approach has been shown to increase the model's resistance to market noise and ensure higher consistency of forecasts at test intervals. The developed reproducible evaluation protocol and defined operating ranges of hyperparameters can be directly implemented in automated decision support systems in the stock markets. The findings expanded the possibilities of using neural networks with parallel-stacked LSTM blocks in financial analysis, forecasting market trends and related AI tasks, as they demonstrate a reproducible configuration with low error under a fixed data preparation and training protocol. The prospects for further research are related to testing the generalisability of the architecture on various assets, frequencies, and forecast horizons, analysing resistance to structural breaks and periods of high-volatility, and investigating the influence of feature composition and integrating attention or ensemble mechanisms for adaptive weighting of time fragments and increasing the robustness of the model.

Acknowledgements
None.

Funding
None.

Conflict of Interest
None.

References

[1] Abbasimehr, H., & Paki, R. (2022). Improving time series forecasting using LSTM and attention models. *Journal of Ambient Intelligence and Humanized Computing*, 13, 673-691. doi: 10.1007/s12652-020-02761-x.

[2] Chicco, D., Warrens, M.J., & Jurman, G. (2021). The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *PeerJ Computer Science*, 7, article number e623. doi: 10.7717/peerj-cs.623.

[3] Draw.io. (n.d.). Retrieved from https://app.diagrams.net/.

[4] Ez-zaiym, M., Senhaji, Y., Rachid, M., El Moutaouakil, K., & Palade, V. (2025). Fractional optimizers for LSTM networks in financial time series forecasting. *Mathematics*, 13(13), article number 2068. doi: 10.3390/math13132068.

[5] Ghojogh, B., & Ghodsi, A. (2023). Recurrent neural networks and long short-term memory networks: Tutorial and survey. *ArXiv*. doi: 10.48550/arXiv.2304.11461.

[6] Gülmez, B. (2023). Stock price prediction with optimized deep LSTM network with artificial rabbits optimization algorithm. *Expert Systems with Applications*, 227, article number 120346. doi: 10.1016/j.eswa.2023.120346.

[7] Kabir, M.R., Bhadra, D., Ridoy, M., & Milanova, M. (2025). LSTM-transformer-based robust hybrid deep learning model for financial time series forecasting. *Sci*, 7(1), article number 7. doi: 10.3390/sci7010007.

[8] Koo, E., & Kim, G. (2022). A hybrid prediction model integrating GARCH models with a distribution manipulation strategy based on LSTM networks for stock market volatility. *IEEE Access*, 10, 34743-34754. doi: 10.1109/ACCESS.2022.3163723.

[9] Kurniawan, A., Indrabayu, & Yusuf, M. (2024). Stock price prediction using technical data and sentiment score. In *2024 IEEE international conference on Industry 4.0, artificial intelligence, and communications technology (IAICT)* (pp. 360-366). Bali: IEEE. doi: 10.1109/IAICT62357.2024.10617768.

[10] Lim, B., Arık, S.O., Loeff, N., & Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4), 1748-1764. doi: 10.1016/j.ijforecast.2021.03.012.

[11] Lindemann, B., Müller, T., Vietz, H., Jazdi, N., & Weyrich, M. (2021). A survey on long short-term memory networks for time series prediction. *Procedia CIRP*, 99, 650-655. doi: 10.1016/j.procir.2021.03.088.

[12] Livieris, I.E., Pintelas, E., & Pintelas, P. (2020). A CNN-LSTM model for gold price time-series forecasting. *Neural Computing and Applications*, 32, 17351-17360. doi: 10.1007/s00521-020-04867-x.

[13] Lytvyn, V., Peleshchak, I., Stepaniak, Y., Peleshchak, R., & Ishchuk, O. (2025). High-precision detection of GPS spoofing attacks on UAVs using MLP. *International Journal of Computing*, 24(2), 254-262. doi: 10.47839/ijc.24.2.4008.

[14] Peleshchak, I., & Futryk, Y. (2025). Time series forecasting using sequentially connected LSTM blocks. Herald of Khmelnytskyi National University. *Technical Sciences*, 347(1), 432-441. doi: 10.31891/2307-5732-2025-347-59.
[15] Sezer, O.B., Gudelek, M.U., & Ozbayoglu, A.M. (2020). Financial time series forecasting with deep learning: A systematic literature review: 2005-2019. *Applied Soft Computing*, 90, article number 106181. doi: 10.1016/j.asoc.2020.106181.
[16] Smyl, S. (2020). A hybrid method of exponential smoothing and recurrent neural networks. *International Journal of Forecasting*, 36(1), 75-85. doi: 10.1016/j.ijforecast.2019.03.017.
[17] Wang, Q., & Zhang, Y. (2022). Research on PM2.5 pollution prediction method in hefei city based on CNN-LSTM hybrid model. *Journal of Physics: Conference Series*, 2400(1), article number 012006. doi: 10.1088/1742-6596/2400/1/012006.
[18] Widiputra, H., Mailangkay, A., & Gautama, E. (2021). Multivariate CNN-LSTM model for multiple parallel financial time-series prediction. *Complexity*. doi: 10.1155/2021/9903518.
[19] Wu, H., Xu J., Wang J., & Longet M. (2021). Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *ArXiv*. doi: 10.48550/arXiv.2106.13008.
[20] Yahoo Finance. (n.d.). *Dataset historical data: GOOG stock price*. Retrieved from https://finance.yahoo.com/quote/GOOG/history/.
[21] Yamak, P.T., Yujian, L., & Gadosey, P.K. (2020). A comparison between ARIMA, LSTM, and GRU for Time Series Forecasting. In *Proceedings of the 2019 2nd international conference on algorithms, computing and artificial intelligence* (pp. 49-55). New York: ACM. doi: 10.1145/3377713.3377722.
[22] Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., & Zhang, W. (2021). Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12), 11106-11115. doi: 10.1609/aaai.v35i12.17325.

Прогнозування часових рядів за допомогою нейромережі з паралельно-стекованими LSTM-блоками

Юрій Футрик

Аспірант
Національний університет «Львівська політехніка»
79000, вул. Степана Бандери, 12, м. Львів, Україна
https://orcid.org/0000-0001-5271-9883

Іван Пелещак

Доктор філософії, доцент
Національний університет «Львівська політехніка»
79000, вул. Степана Бандери, 12, м. Львів, Україна
https://orcid.org/0000-0002-7481-8628

Анотація. Прогнозування часових рядів є критично важливими для підтримки рішень у фінансовій аналітиці, де дані характеризуються нелінійністю, нестаціонарністю та високим рівнем шуму. Метою роботи було визначення ефективної конфігурації рекурентної нейронної мережі з паралельним поєднанням стеків осередків Long Short-Term Memory (LSTM) для підвищення точності прогнозування цін акцій, а також можливостей застосування прикладної моделі в галузі промисловості, енергетиці та суміжних доменах. У дослідженні застосовано методи глибинного навчання з використанням бібліотеки TensorFlow/Keras, а для навчання моделі використано історичні дані акцій корпорації Google. Встановлено, що архітектура з паралельно-стекованими блоками забезпечує вищу стабільність навчання порівняно зі стандартними рекурентними моделями за рахунок ефективнішого виділення технічних ознак часової послідовності. Експериментально доведено, що оптимальна кількість нейронів у прихованих шарах для такої задачі становить 100-200 одиниць, тоді як подальше збільшення потужності моделі призводить до ефекту перенавчання. Виявлено, що застосування регуляризації Dropout у діапазоні 0,1-0,2 дозволяє мінімізувати помилку на валідаційній вибірці, у той час як значення понад 0,3 суттєво уповільнюють збіжність алгоритму. Аналіз інженерії ознак показав, що інтеграція експоненціального ковзного середнього з коротким вікном часу покращує результат моделі, демонструючи вищу кореляцію з цільовим показником, ніж індекс відносної сили. Якість прогнозування моделі оцінювали за середньоквадратичною помилкою (Mean Squared Error, MSE), коренем із неї (Root Mean Squared Error, RMSE) і середньою абсолютною відсотковою похибкою (Mean Absolute Percentage Error, MAPE). Встановлено, що для конфігурацій (50-100 блоків) характерні підвищені значення MAPE, тоді як у діапазоні 180-400 блоків похибка зменшується та набуває стабільного характеру. Найточніший результат отримано для конфігурації з 325 блоками, регуляризацією Dropout = 0,05 та оптимізатором Nadam (Nesterov-accelerated Adam): MAPE = 1,62 %, RMSE = 2,41, MSE = 6,05. Практична цінність дослідження полягає у формулюванні чітких рекомендацій щодо налаштування гіперпараметрів LSTM-моделей для прикладного короткострокового прогнозування фінансових рядів

Ключові слова: глибинне навчання; Dropout-регуляризація; Nadam-оптимізація; EMA; RSI