EAI Endorsed Transactions
on Tourism, Technology and Intelligence
Research Article EAI.EU

GS-MultiRC: Multi-step Reservoir Computing Leveraging Grid Search for Stock Indices Prediction

Quan Thanh Dao[1] and Chau Bao Phung[1]

[1]Memorial University, Canada

Abstract

Stock market prediction plays a crucial role in investment decision-making, portfolio management, and risk assessment, significantly impacting financial stability and economic growth. Accurately forecasting stock prices, which are chaotic and nonlinear, has become a main point of financial research. Deep learning approaches, such as neural networks and long-short-term memory (LSTM) models, have been more reliable than traditional approaches such as the ARMA and ARIMA models. However, these methods require a lot of computational power, complex fine-tuning procedures, and often overfit, especially with limited or noisy data. Reservoir Computing (RC) has emerged as a potential alternative for financial time series prediction. It uses a fixed, randomly connected reservoir to capture patterns in data, requiring only the output layer to be trained. This design makes RC computationally efficient and simpler to use. However, RC models can struggle with overfitting when the reservoir is too large compared to the data or when the model can not adapt well to unseen data. To address these drawbacks, we propose a multi-step RC model, focusing on popular stock indices, including CSI300, FTSE100, S&P500, and SSE50. Our approach includes a retraining step where the reservoir evolves by forecasting some of the training data and simulating real-world testing conditions. These evolved internal states, affected by prediction errors, are used to retrain the output layer, making the model more robust and less likely to overfit. Our experiments show that our model performs more accurately and efficiently than conventional RC and LSTM models, making it a workable and trustworthy option for stock market prediction. This work contributes to utilizing RC-based approaches in terms of the financial forecasting domain.

Keywords: Reservoir computing, Time series prediction, Stock price prediction

Received on 30 April 2025; accepted on 14 July 2025; published on 04 August 2025

doi:10.4108/eettti.9200

1. Introduction

Stock market prediction is crucial for making smart investment choices, managing assets effectively, and identifying potential risks, as it influences financial stability and economic growth. Accurately forecasting stock prices, known as chaotic and nonlinear, has become a key focus in financial r esearch [1][2]. Recently, improvements in machine learning have enhanced stock price prediction, providing robust tools to solve this challenging task. For example, neural networks and long-short term memory (LSTM) networks approaches have shown stronger performance over conventional statistical predictive models such as auto-regressive moving average model (ARMA) and auto-regressive integrated moving average model (ARIMA) in predicting financial time series [3][4]. Although these models are highly predictive, they often encounter issues such as high computational resources, complex hyper-parameter tuning, and vulnerability to overfitting, especially when dealing with limited data or noisy inputs [5].

As an alternative, Reservoir Computing (RC) has recently gained interest in the financial field. Originally based on the concept of recurrent neural networks, RC leverages a fixed, randomly created reservoir layer with sparse connections to capture complex dynamics, requiring only the output layer to be trained. This architecture offers significant computational efficiency, as the fixed input layer and internal reservoir eliminate

*Corresponding author. Email: qtdao@mun.ca

the need for back-propagation training throughout the network [6][7]. There are multiple motivations for applying RC to stock data. For example, Wang et al. [8] showed the outperformance of RC with EMD2FNN, which was developed by integrating empirical mode decomposition with a neural network and ModAugNet (LSTM model with overfitting prevention module) on various indices. Tian et al. [9] applied sparrow search to optimize the RC models. The results are compared with other deep learning models and show the RC model's superiority. Wang et al. [5] proposed a compressed sensing method to improve the connectivity between nodes within the internal states of the RC model. Consequently, the model was applied to the financial data. These contributions have pushed the potential of applying RC in the financial domains.

In practice, the RC model has to have significant memory to capture all the necessary patterns of indices. The memory capacity can be improved by adding more nodes to the reservoir, meaning increasing its size. However, some studies have shown that increasing the reservoir size only improves RC performance up to a certain point in various computational tasks [10][11]. Moreover, if the reservoir size significantly exceeds the length of the input time series data, it can cause an "ill-posed problem" during the training of the linear readout layer [12][13], leading to the RC not being able to predict the data correctly.

A possible solution to overcome the aforementioned limitations is to use multiple reservoirs (as known as multi-step reservoir models). In this work, inspired by the multi-reservoir approaches [14][15], we develop a multi-step RC model tailored for financial time series prediction, focusing on four popular stock indices, such as CSI300, FTSE100, S&P500, and SSE50. Our proposed model introduces a retraining mechanism to mitigate overfitting commonly observed in standard RC models. After an initial training phase, the reservoir's internal states are evolved by forecasting a portion of the training data, allowing the system to replicate the dynamics encountered during testing. This strategy captures the impact of small prediction errors on internal states, which are later used to update the readout layer. Furthermore, we employ grid search to identify the optimal set of hyperparameters, ensuring efficient model performance. By demonstrating the reservoir's sensitivity to prediction errors and retraining accordingly, our method improves generalization and reduces test errors. We also compare the predictive performance of our model against traditional RC and LSTM models, showcasing both effectiveness and efficient resource usage. This work contributes to utilizing RC-based approaches in terms of the financial forecasting domain.

2. Methodology

2.1. Reservoir computing model

Figure 1.a illustrates the structure of a reservoir computing (RC) model. At each time step t, the model includes:

- Input vector represents K input features.

$$\vec{u}(t) = (u_1(t), u_2(t), \ldots, u_K(t)) \tag{1}$$

- Internal states vector represents the N-node network's dynamic reservoir.

$$\vec{x}(t) = (x_1(t), x_2(t), \ldots, x_N(t)) \tag{2}$$

- Output vector represents predictions of L features.

$$\vec{y}(t) = (y_1(t), y_2(t), \ldots, y_L(t)) \tag{3}$$

Assuming $t = 1, 2, .., T$ where T is the data length. The reservoir state updates at each step are defined by the following equation:

$$\vec{x}(t) = (t - \alpha)\vec{x}(t-1) + \alpha\vec{\tilde{x}}(t) \tag{4}$$

where

$$\vec{\tilde{x}}(t) = f\left(W^{\text{in}}\vec{u}(t) + W\vec{x}(t-1) + W^{\text{back}}\vec{y}(t-1)\right) \tag{5}$$

Where $W^{\text{in}} \in R^{N\times K}$, $W \in R^{N\times N}$, $W^{\text{back}} \in R^{N\times L}$ are the weights connecting the input to the internal states, the connections between nodes inside the reservoir, and weights from the output to the reservoir, respectively. All these weights are randomly initialized and remain unchanged throughout the training phase. Besides, $f(.)$ can represent nonlinear activation functions, such as $tanh(.)$ or $ReLU(.)$. The leaking rate $\alpha \in (0, 1]$ is used to enhance the effect of previous states on the current state. This value might help enhance the memory capacity of the internal network. The output of the network is computed using the following formula:

$$\vec{y}(t) = f^{out}(W^{out}\vec{x}(t)) \tag{6}$$

where $W^{\text{out}} \in R^{L\times N}$, which is the only one trained in the network, denotes the readout weights and is obtained by least-square regression with Ridge regularization (regularization parameter γ) to prevent overfitting. With $X = (x(1), x(2), ..., x(T)) \in R^{N\times T}$, $Y = (y(1), y(2), \ldots, y(T)) \in R^{L\times T}$ are the set of internal states, and labels in each discrete time respectively, the optimized values of W are archived by:

$$W^{out} = YX^T(XX^T + \gamma I)^{-1} \tag{7}$$

In RC, the echo state condition is an important requirement [6]. It depends on the spectral radius,

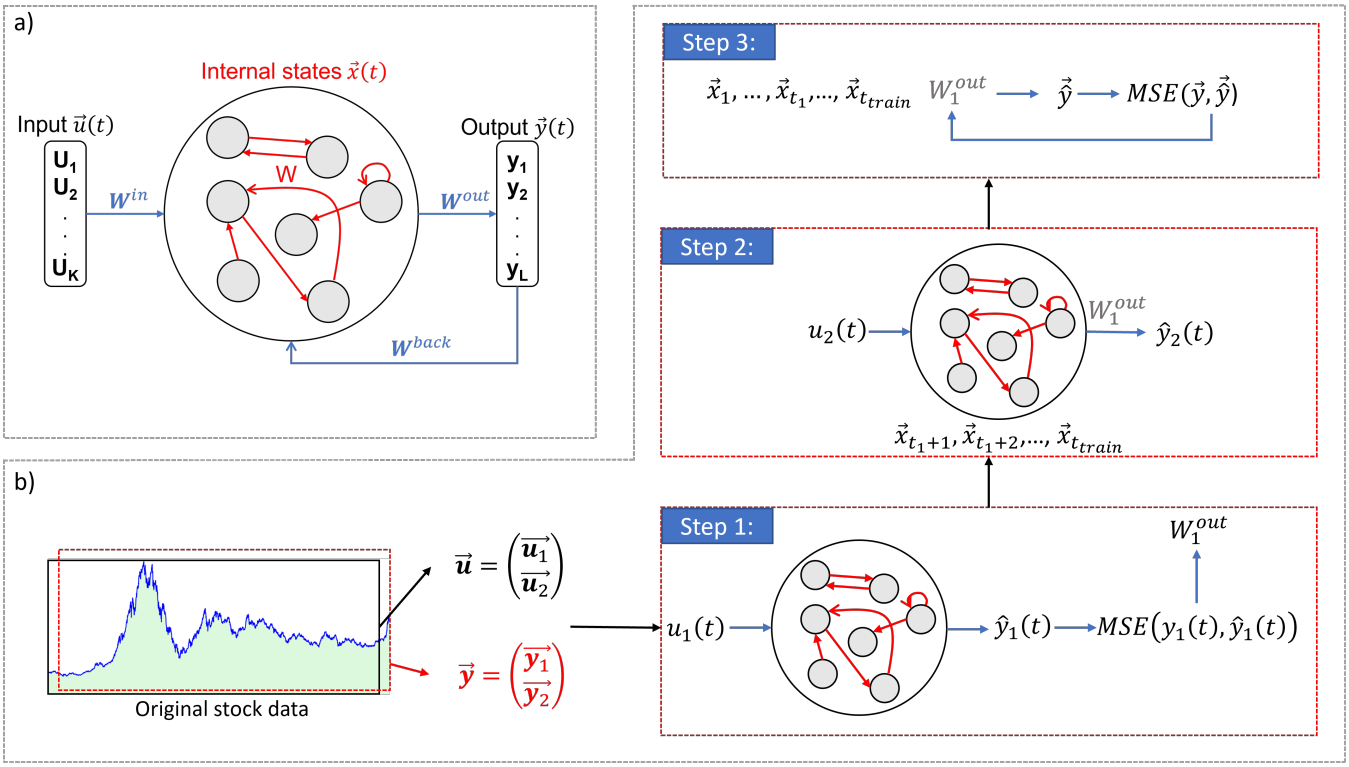

Figure 1. a. The architecture of a traditional reservoir computing model, b. Proposed model for financial data.

ρ(W) of the reservoir weight matrix, W. The spectral radius is the largest absolute value among the matrix's eigenvalues and gives a rough idea of how much memory the reservoir can retain. A smaller spectral radius means the reservoir remembers things for a short time, while a larger one allows for longer memory. Nevertheless, when the spectral radius grows too large, the system can lose stability, and the echo state condition breaks down [16]. Besides, the sparsity value, referring to how many connections between nodes in the reservoir are set to zero, is also used to enhance the model's performance [17].

2.2. The proposed multi-step reservoir computing model

When the standard RC with linear readout layer overfits the training data, even small changes in the internal states caused by prediction errors can lead to large prediction mistakes [18]. In our proposed method, during the initial training phase, we let the internal states change by predicting part of the training data, thereby replicating the testing scenario where prediction errors naturally occur and accumulate. These errors are passed to the internal states and then used to update the readout layer. This learning approach shows how small prediction errors influence the internal states during the testing phase. This effect significantly reduces the test errors by reducing overfitting. The training strategies are shown as:

- Step 1: Split the training data into two parts.

$$\vec{\boldsymbol{u}} = \begin{bmatrix} \vec{u}[1] \\ \vec{u}[2] \\ \vdots \\ \vec{u}[t_1] \\ \vec{u}[t_1+1] \\ \vdots \\ \vec{u}[t_{\text{train}}] \end{bmatrix} = \begin{bmatrix} \vec{u}_1[1] \\ \vec{u}_1[2] \\ \vdots \\ \vec{u}_1[t_1] \\ \vec{u}_2[1] \\ \vdots \\ \vec{u}_2[t_{\text{train}}] \end{bmatrix} = \begin{pmatrix} \vec{\boldsymbol{u}}_1 \\ \vec{\boldsymbol{u}}_2 \end{pmatrix}, \tag{8}$$

$$\vec{\boldsymbol{y}} = \begin{bmatrix} \vec{y}[1] \\ \vec{y}[2] \\ \vdots \\ \vec{y}[t_1] \\ \vec{y}[t_1+1] \\ \vdots \\ \vec{y}[t_{\text{train}}] \end{bmatrix} = \begin{bmatrix} \vec{y}_1[1] \\ \vec{y}_1[2] \\ \vdots \\ \vec{y}_1[t_1] \\ \vec{y}_2[1] \\ \vdots \\ \vec{y}_2[t_{\text{train}}] \end{bmatrix} = \begin{pmatrix} \vec{\boldsymbol{y}}_1 \\ \vec{\boldsymbol{y}}_2 \end{pmatrix}. \tag{9}$$

- Step 2: In the first training part, the internal reservoir states $X_1 \in R^{N \times t_1}$ are computed, followed by applying ridge regression to determine the output weights W_1^{out}.

- Step 3: Evolve the internal states of the reservoir $X_2 \in R^{N \times t_2}$ by letting the reservoir predict the values of $\vec{y}_2$.
- Step 4: The readout layer is retrained using all internal states $X \in R^{N \times (t_1+t_2)}$ along with the full training data $\vec{y}$ to optimize a new value for W^{out}.

Regarding the stock indices data, the price of the current step is the input for the next day. Figure. 1.b provides a visual overview of the proposed model designed for stock data.

2.3. Assessment metrics

We evaluate our model's performance using several standard metrics: mean absolute percentage error (MAPE), mean absolute error (MAE), root mean square error (RMSE), symmetric mean absolute percentage error (SMAPE), and theil inequality coefficient (TIC). Besides, to depict the fluctuation, we also use trend direction accuracy (DA), to capture the model's ability to track trend changes [19]. DA is defined in the following equation:

$$DA(d, y) = \frac{\sum_{k=1}^{N-1} F(\text{sgn}(d_{k+1} - d_k) - \text{sgn}(y_{k+1} - y_k))}{N}, \tag{10}$$

where d_k indicates the predicted value, y_k represents the true value, and N denotes the total number of samples in the target sample set. The function $sgn(x)$ is the sign function. Both $F(x)$ and $sgn(x)$ are defined as follows:

$$F(x) = \begin{cases} 1 & \text{if } x = 0, \\ 0 & \text{if } x \neq 0. \end{cases} \tag{11}$$

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{if } x = 0, \\ -1 & \text{if } x < 0. \end{cases} \tag{12}$$

Smaller MAE, MAPE, RMSE, and SMAPE values and a TIC value closer to 0 show higher prediction accuracy and smaller errors. Conversely, larger MAE, MAPE, RMSE, and SMAPE values, with a TIC value closer to 1, reflect lower accuracy. Besides, a higher DA value suggests better alignment between the predicted and actual trends, enhancing the model's ability to accurately capture upward and downward movements in stock prices.

3. Experiments and Results

3.1. Experiment settings

To assess the performance of the proposed model, experiments were conducted using daily opening prices of four popular datasets (CSI300, FTSE100, S&P500, SSE50) sourced from different regions (US, UK, China), ensuring a diverse representation of data characteristics. The datasets cover the period from 2004 to 2024 and include a total of 19472 data points. The data was divided into training and testing subsets, where 80% allocated for training and the remaining 20% used for testing. Table 1, including key statistics, provides a summary of the datasets.

As mentioned, the training process was carried out in two phases. In the first phase, 75% of the training data was used to optimize the model's output weight W^{out}. The remaining 25% of the training data was then used to evolve the internal states X_2. To ensure the best performance of the model, a grid search was performed to find the optimal values for the key hyperparameters, including spectral radius ρ(W), sparsity, reservoir size N, and warm-up time t_{min}. The ranges and step sizes for these parameters are detailed in Table 3.

To evaluate performance, the proposed model was compared with two commonly used baseline models: the traditional RC model and the LSTM network. Since each market index represents a separate prediction task, the experiments were performed independently for each one. Accordingly, a distinct RC model was constructed, trained, and evaluated for every index, and the same process was followed for the LSTM model to maintain a fair comparison. The accuracy and robustness of the models were evaluated using metrics stated in Section 2.3.

3.2. Results

Figure 2 shows the predictions in the test data of the proposed model, traditional RC model, and LSTM model. Aside from applying grid search for the proposed model, we also used that method to find the best-performing traditional RC model. In terms of the LSTM model, we repeated the training process by changing configurations such as the number of epochs, size of hidden layers, etc. to achieve the highest accuracy. Intuitively, while the LSTM model can only simulate the patterns of the data, traditional RC and our model fit the actual prices very well. Furthermore, all methods seem hard to fit the FTSE100 data in the period after April 2024.

More specifically, Figure 3 displays the absolute percentage errors (APE) of the predicted stock prices generated by our model. As shown, these error values remain below 1% across all indices, indicating that the predicted prices closely match the actual values with minimal errors. Moreover, the APE values in 2022 of the S&P500 show higher values than average, and all errors show sudden high values after 2024.

Finally, Table 2 calculates the performance of models in the stated metrics. The results show that the

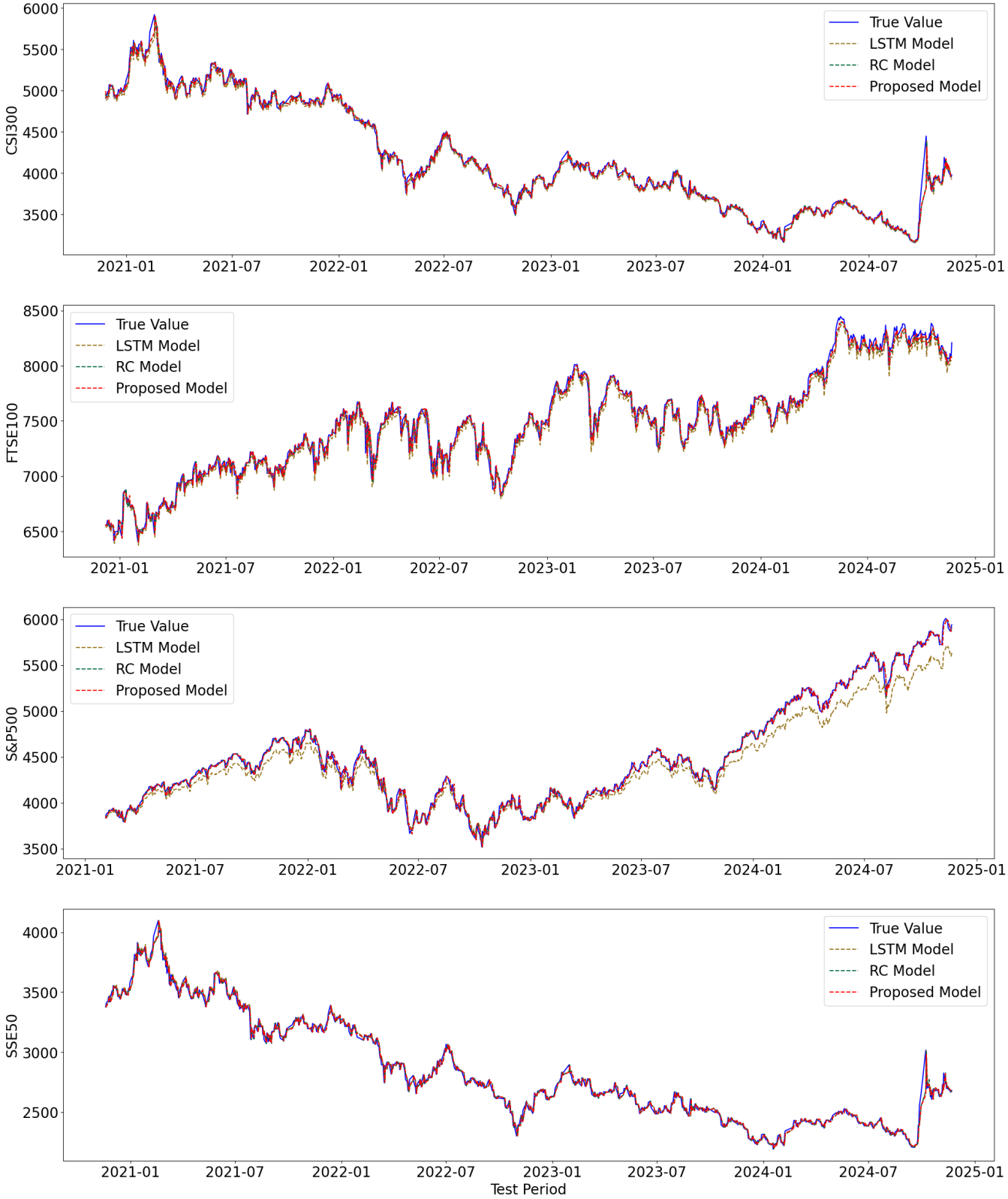

Figure 2. Plots of predicted index prices of LSTM, traditional RC, proposed model, and ground truth price for each stock market index in the test period. The predicted prices result from the best-performing models obtained in the training period.

Table 1. Summary statistics for various indices.

Index	Study Period (Start-End)	Total Number	Training Size	Testing Size	Min	Max	Mean	Std. Dev
CSI300	2005/01/04–2024/11/21	4833	3866	967	816.55	5922.07	3237.69	1080.64
FTSE100	2005/02/21–2024/11/22	4991	3992	999	3512.10	8445.80	6444.75	949.60
S&P500	2005/11/21–2024/11/21	4783	3826	957	679.30	6008.86	2387.79	1256.31
SSE50	2004/11/22–2024/11/22	4865	3892	973	699.27	4726.08	2281.72	754.41

Table 2. Best performances for individual index achieved by different network structures and settings. Three types of networks are LSTM, traditional RC, and proposed model.

Index	Model	MAE	MAPE	RMSE	SMAPE	TIC	AD
CSI300	LSTM	43.876	1.022	62.471	1.023	0.0074	0.484
	Traditional RC	38.254	0.901	55.048	0.903	0.0065	0.477
	Proposed model	**37.994**	**0.895**	**54.728**	**0.897**	**0.0065**	**0.472**
FTSE100	LSTM	58.923	0.786	74.347	0.789	0.0049	0.484
	Traditional RC	44.308	0.597	61.085	0.597	0.0041	0.490
	Proposed model	**43.996**	**0.593**	**60.595**	**0.593**	**0.0040**	**0.484**
S&P500	LSTM	108.085	2.257	133.724	2.291	0.0149	0.527
	Traditional RC	31.725	0.722	42.409	0.722	0.0047	0.523
	Proposed model	**31.701**	**0.722**	**42.408**	**0.722**	**0.0047**	**0.524**
SSE50	LSTM	25.809	0.894	37.304	0.894	0.0065	0.475
	Traditional RC	25.856	0.897	36.871	0.898	0.0064	0.467
	Proposed model	**25.486**	**0.884**	**36.608**	**0.885**	**0.0064**	**0.476**

Table 3. Grid Search Configurations

Parameter	Range
Spectral radius (ρ)	0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3
Sparsity	0.75, 0.8, 0.85, 0.9, 0.95
Reservoir size	50, 100, 150, 200, 250, 300, 350, 400
Warm-up $t_{\min}$	0, 50, 100

proposed method significantly outperforms the LSTM methods, and slightly outperforms traditional RC model.

4. Conclusion

This study explores the potential of a multi-step Reservoir Computing (RC) model for predicting stock market trends, focusing on the challenges posed by chaotic and nonlinear financial time series. By introducing a retraining mechanism and leveraging a grid search for hyperparameter optimization, the proposed model enhances predictive accuracy while mitigating issues of overfitting and resource inefficiency commonly observed in traditional RC and LSTM models.

The experimental results on four popular stock indices — CSI300, FTSE100, S&P500, and SSE50 — highlight the proposed model's accuracy and robustness. The ability to capture the dynamics

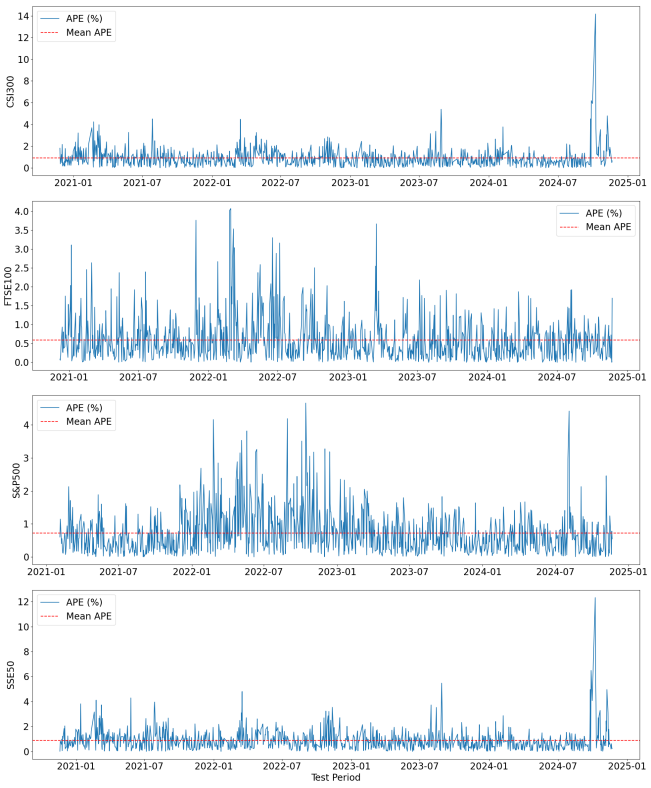

Figure 3. Absolute percentage error (APE) for each stock market index during the test period using the proposed model.

of financial data more effectively is highlighted through consistently lower absolute percentage errors (APE) and improved performance metrics across most datasets compared to the baseline models. Notably, the retraining strategy enabled the model to adapt to prediction errors and improve its generalization, as evidenced by its ability to handle challenging time periods where other models struggled, such as the FTSE100 data after April 2024.

By showcasing the efficiency and scalability of the multi-step RC approach, this work adds valuable insights to the growing body of research on applying RC in financial forecasting. The outcomes show that the model delivers both high prediction accuracy and strong computational performance, making it highly suitable for real-world scenarios where speed and precision are critical. Moreover, the work opens up exciting possibilities for future research, such as exploring multi-reservoir models that could further enhance performance and be adapted to other fields with complex time series dynamics.

Acknowledgment

This work was supported by the Canada Excellence Research Chair (CERC) Program CERC-2022-00109.

References

[1] W. Jiang, "Applications of deep learning in stock market prediction: recent progress," *Expert Systems with Applications*, vol. 184, p. 115537, 2021.

[2] M. M. Kumbure, C. Lohrmann, P. Luukka, and J. Porras, "Machine learning techniques and data for stock market forecasting: A literature review," *Expert Systems with Applications*, vol. 197, p. 116659, 2022.

[3] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PloS one*, vol. 12, no. 7, p. e0180944, 2017.

[4] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European journal of operational research*, vol. 270, no. 2, pp. 654–669, 2018.

[5] Z. Wang, H. Zhao, M. Zheng, S. Niu, X. Gao, and L. Li, "A novel time series prediction method based on pooling compressed sensing echo state network and its application in stock market," *Neural Networks*, vol. 164, pp. 216–227, 2023.

[6] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, vol. 148, no. 34, p. 13, 2001.

[7] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural computation*, vol. 14, no. 11, pp. 2531–2560, 2002.

[8] W.-J. Wang, Y. Tang, J. Xiong, and Y.-C. Zhang, "Stock market index prediction based on reservoir computing models," *Expert Systems with Applications*, vol. 178, p. 115022, 2021.

[9] Z. Tian and L. Pei, "Stocks price prediction based on optimized echo state network by sparrow search algorithm," *International Journal of Dynamics and Control*, pp. 1–14, 2024.

[10] J. Pathak, A. Wikner, R. Fussell, S. Chandra, B. R. Hunt, M. Girvan, and E. Ott, "Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 28, no. 4, 2018.

[11] A. Chattopadhyay, P. Hassanzadeh, and D. Subramanian, "Data-driven predictions of a multiscale lorenz 96 chaotic system using machine-learning methods: reservoir computing, artificial neural network, and long short-term memory network," *Nonlinear Processes in Geophysics*, vol. 27, no. 3, pp. 373–389, 2020.

[12] T. Akiyama and G. Tanaka, "Computational efficiency of multi-step learning echo state networks for nonlinear time series prediction," *IEEE Access*, vol. 10, pp. 28 535–28 544, 2022.

[13] M. Lukoševičius, "A practical guide to applying echo state networks," in *Neural Networks: Tricks of the Trade: Second Edition*. Springer, 2012, pp. 659–686.

[14] F. Triefenbach, A. Jalalvand, B. Schrauwen, and J.-P. Martens, "Phoneme recognition with large hierarchical reservoirs," *Advances in neural information processing systems*, vol. 23, 2010.

[15] T. Akiyama and G. Tanaka, "Analysis on characteristics of multi-step learning echo state networks for nonlinear time series prediction," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–8.

[16] N. Trouvain, L. Pedrelli, T. T. Dinh, and X. Hinaut, "Reservoirpy: an efficient and user-friendly library to design echo state networks," in *International Conference on Artificial Neural Networks*. Springer, 2020, pp. 494–505.

[17] F. M. Bianchi, S. Scardapane, S. Løkse, and R. Jenssen, "Reservoir computing approaches for representation and classification of multivariate time series," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 5, pp. 2169–2179, 2020.

[18] L. Domingo Colomer, "Classical and quantum reservoir computing: development and applications in machine learning," Ph.D. dissertation, Agronomica, 2023.

[19] C. Wang, Y. Yang, L. Xu, and A. Wong, "A hybrid model of primary ensemble empirical mode decomposition and quantum neural network in financial time series prediction," *Fluctuation and Noise Letters*, vol. 22, no. 04, p. 2340006, 2023.