Efficient Application of Multi-Layer Data Processing with PBFT and RAFT for Transaction Forecasting Using LSTM

Alexander Bogdanov[1], Jasur Kiyamov[1,4(✉)], Valery Khvatov[2], Gennady Dik[1,3], Aleksandr Dik[1,3], Egor Savkov[5], and Aleksandr Shchegolev[6]

[1] St. Petersburg University, St. Petersburg, Russia
{a.v.bogdanov,z.kiyamov,a.dik}@spbu.ru, g.dick@systechnologies.ru
[2] DGT Technologies AG, Montreal, Canada
[3] St. Petersburg LLC "System Technologies", St. Petersburg, Russia
[4] Samarkand branch of Tashkent University of Information Technologies, Samarkand, Uzbekistan
[5] Consern Avrora Scientific and Production Association JSC, St. Petersburg, Russia
[6] St. Petersburg State Marine Technical University, St. Petersburg, Russia

Abstract. The article considers approaches to multi-level data processing using the Practical Byzantine Fault Tolerance (PBFT) and Reliable, Replicated, and Fault-Tolerant (RAFT) consensus algorithms. Their application for organizing efficient processing and aggregation of transactional data in distributed systems is analyzed. The main focus is on the use of recurrent neural networks with long short-term memory (LSTM) for predicting transactions based on data processed by PBFT and RAFT. The results of experiments on improving the accuracy of predictions taking into account various consensus strategies are presented.

Keywords: LSTM · PBFT · RAFT · Blockchain · Consensus Algorithms · Deep Learning

1 Introduction

Accurate forecasting of transactions in a distributed system is crucial in the maintenance of reliability and security [1]. Most contemporary financial and business systems require accuracy on the order of more than 90%, as any inaccuracies may cause users to lose a lot of money and confidence in the organization. Therefore, there is a need to employ information processing methods that not only guarantee data integrity and consistency, but can also aid in predicting future transactions with a high level of accuracy by detecting obscured patterns. An automation method for transaction data issues are the consensus algorithms, like PBFT and RAFT, which guarantee consistency in the presence of a partially reliable network [2,3].

PBFT offers protection from some interfering attacks and system failures, while RAFT makes the consensus process easier and more intuitive. Yet, these methods are insufficient to precisely predict transaction dynamics. The adoption

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026
O. Gervasi et al. (Eds.): ICCSA 2025 Workshops, LNCS 15887, pp. 103–114, 2026.
https://doi.org/10.1007/978-3-031-97589-9_8

of some computer science techniques with particular methods, like the integration of LSTM type neural networks, can noticeably enhance forecasting accuracy.

LSTM networks have proven to be one of the most, if not the single most efficient and effective transactional time series tool alongside with their counterparts. Their capability to account for long-term dependencies makes them indispensable when dealing with predicted noise from distributed systems. This paper demonstrates the advantages of blending.

2 PBFT and RAFT Consensus Algorithms

Two well-liked consensus algorithms that guarantee dependable operation of distributed systems are PBFT and RAFT (Table 1):

- By permitting up to one-third of nodes to be malicious, PBFT (Practical Byzantine Fault Tolerance) is made to function in the face of potential attacks and failures;
- Implementation simplicity is the main goal of RAFT (Reliable, Replicated, and Fault-Tolerant), which is utilized in less hostile environments [4].

These algorithms guarantee high confidence in the input data for the LSTM model by enabling data to be compiled and arranged prior to being provided into a machine learning system [5].

Table 1. Comparison table of PBFT and RAFT

Characteristic	PBFT	RAFT
Failure type	Byzantine (up to 1/3 of nodes may be malicious)	Node failures (only crashes, no malicious behavior)
Operating speed	Slower due to complex approval stages	Faster because it requires fewer messages
Computational costs	Hight	Low
Implementation complexity	Hight	Low
Application area	Highly secure financial systems, blockchains	Cloud services, distributed databases

For distributed systems to be dependable, secure, and resilient to possible failures and anomalies, transaction forecasting is essential. High prediction accuracy is necessary for contemporary business, blockchain, and financial applications since mistakes can result in large losses, erode user trust, and cause the system to become unstable. Transaction forecasting is difficult because of the unpredictability of data flows, noise and anomalous values, and reliance on numerous outside variables like market fluctuations and cyberattacks.

Complex data processing techniques that guarantee the integrity, consistency, and dependability of information are required to produce forecasts with the highest possible accuracy and dependability. The PBFT and RAFT consensus algorithms, which remove duplication, filter out anomalies, and arrange transactions to produce structured time sequences, are crucial to this process. Recurrent neural networks (LSTM) can be used to further analyse this data in order to find deep patterns, model recurrent patterns, and create precise time series forecasts.

When PBFT, RAFT, and LSTM are used together, the model is more resilient to noise, anomaly detection is more effective, and the likelihood of inaccurate predictions is decreased. This method can be used in distributed computing, cybersecurity, blockchain data analysis, and financial transactions. Neural network architecture optimisation, model adaptation to shifting market conditions, and integration with hybrid machine learning techniques are possible future developments of this methodology.

One of the best methods for processing time series, including transaction flows, has been shown to be LSTM networks [6]. They are the best option for transaction prediction tasks in distributed systems because of their capacity to filter noise and take long-term dependencies into account [7]. The advantages of combining PBFT and RAFT with LSTM for transaction data prediction are covered in this paper, along with an analysis of experimental findings that support the efficacy of the suggested methodology. The work algorithm PBFT → RAFT → LSTM (Fig. 1)

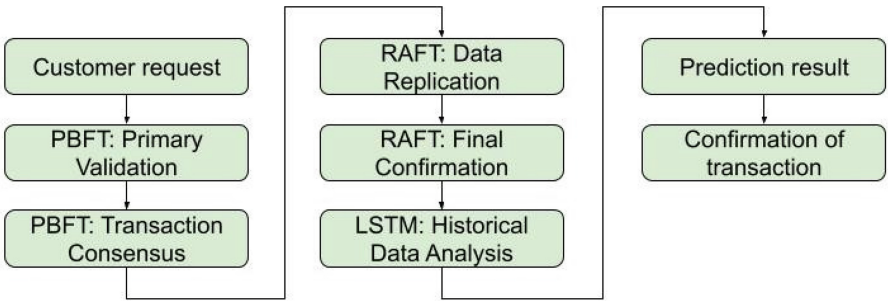

Fig. 1. Block diagram PBFT → RAFT → LSTM

1. PBFT: Attack Defences and Initial Validation
   - To complete a transaction, the client submits a request.
   - PBFT confirms the data's consistency and authenticity.
   - The following stages are carried out: Prepare → Commit → Prepare
   - RAFT receives the consistent data
2. RAFT: Fast Replication and Final Confirmation
   - RAFT receives data validated by PBFT
   - RAFT leader distributes data among nodes

   - A majority of nodes confirms the transaction commit
   - The finalized data is passed to LSTM
3. LSTM: Forecasting Upcoming Deals
   - Data that has been cleaned and replicated is sent to LSTM.
   - The time series of transactions is analysed by the trained model.
   - LSTM uses historical data to forecast future transactions.
4. The result's output
   - The network is optimised using the predicted data.
   - The client is given a forecast for upcoming operations as well as details regarding the execution of the transaction.

3 Applying LSTM to Transaction Prediction

When it comes to time series forecasting, including transaction flow analysis, recurrent neural networks—particularly their enhanced LSTM model—show excellent efficiency. They are among the most dependable tools for predictive analysis in distributed and financial systems because of their capacity to account for long-term dependencies, eliminate noisy outliers, and adjust to dynamic data changes. However, the quality of the input data, which necessitates preprocessing using trustworthy consensus mechanisms, has a significant impact on forecast accuracy. You can improve the consistency of transaction flows, remove inconsistencies, and filter out anomalies by using PBFT and RAFT before feeding data to the LSTM model (Fig. 2). RAFT ensures data ordering and replication, which lowers the amount of noise in the data, while PBFT ensures protection against Byzantine faults. This lowers the possibility of model errors and greatly enhances the quality of predictions. The following steps are involved in the transaction forecasting process using PBFT, RAFT, and LSTM:

1. Gathering transaction data and pre-processing it with RAFT and PBFT to get rid of inaccurate information.
2. Establishing time sequences and normalising data, which enhances model convergence and lessens the influence of outliers.
3. Optimising the network architecture, number of layers, training window size, and gradient descent parameters when training the LSTM model.
4. To avoid overfitting and improve the model's resistance to anomalies, regularisation (Dropout, L2) is used.
5. Examining the precision of forecasts with the MSE (Mean Squared Error), RMSE (Root Mean Squared Error) and MAE (Mean Absolute Error) measures lets one assess their quality.
6. Results visualisation and testing on actual transaction data let you customise the model to the particular system characteristics.

Since the model operates on structured and pre-agreed data, combining PBFT and RAFT with LSTM improves the dependability and accuracy of predictions. Forecasting mistakes can cause major financial losses in financial platforms, blockchain systems, and distributed computing environments, making this

strategy particularly pertinent. Using convolutional neural networks (CNN) to emphasise complex data patterns or implementing hybrid models like LSTM + Transformer could help to increase accuracy even more in the future. Therefore, the combination of consensus algorithms and a deep neural network opens up new possibilities for exact analysis of transaction flows and the development of smart predictive systems.

Recurrent neural networks (RNNs), especially their advanced version LSTM (Long Short-Term Memory), are a powerful tool for time series analysis. In the context of transaction forecasting, LSTM allows you to identify patterns and predict future transactions based on historical data [8].

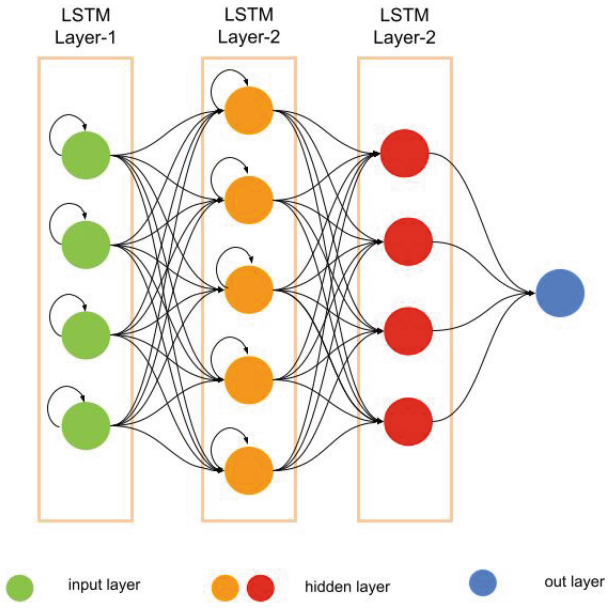

Fig. 2. LSTM architecture

Because PBFT and RAFT consensus algorithms guarantee the consistency and purity of the input data, using pre-processed data greatly increases prediction accuracy.

3.1 Key Steps of LSTM Application in Transaction Forecasting

Filtering, normalising, and structuring time sequences are the goals of the multi-stage processing that the data goes through before being fed into the LSTM model. By doing this, the noise level is decreased, mistakes are removed, and the model's predictions are guaranteed to be highly accurate. Pre-processing's primary objective is to guarantee data consistency and dependability, which is

crucial in distributed systems where transactions could be vulnerable to failures or attacks.

1. **Transaction data collection and preprocessing (PBFT and RAFT)**
   Prior to analysis, the PBFT and RAFT consensus algorithms are combined to increase the consistency and dependability of the data. These algorithms improve the stability of transaction data by removing errors brought on by malevolent nodes, network outages [9,10], or random anomalies.
   - By detecting malicious or incorrect transactions, PBFT removes Byzantine errors and offers defence against attacks.
   - Transactional data is organised and replicated by RAFT, which guarantees its synchronisation and integrity amongst system nodes.

   These steps lessen the possibility of noise, duplicates, and inconsistencies by providing LSTM with pre-processed, cleaned, and trustworthy transactional data. Predictions become more accurate and stable as a result, and model convergence is greatly enhanced.
2. **Normalising data and creating time sequences**
   The data is organised and ready for training following pre-processing, which consists of two crucial steps:
   - Data normalisation speeds up model training and lessens the effect of various data scales on the network weight optimisation process by bringing all numerical values into the range [0,1] or [-1,1].
   - creation of time sequences: every transaction is shown as a window of context for earlier occurrences. For instance, when utilising a window of size 10, the model predicts the next transaction after analysing the previous ten.

The removal of anomalous data found through statistical analysis is one of the additional data processing enhancements:

- incorporating a sliding window to take transaction flow dynamics into consideration.
- increasing the training set's diversity through the use of data augmentation techniques.

Transaction data is subjected to deep filtering, normalisation, and structuring through the combined application of PBFT, RAFT, and LSTM. This reduces forecast errors, boosts the model's resilience to noise, and enhances prediction accuracy. This method works particularly well in decentralised, blockchain, and financial systems where transaction accuracy and consistency are crucial (Table 2).

LSTM is trained on the previous values (T-3, T-2, T-1) to predict the next value (T-n).

Table 2. Example of preparing input data for LSTM

Time	Transaction amount	Number of inputs	Number of inputs
T-3	0.32	2	2
T-2	0.45	3	3
T-1	0.38	1	1
T-n	...	...	...

3.2 Training an LSTM Model with Optimized Hyperparameters

Transaction prediction is essential to the efficiency, security, and dependability of contemporary distributed systems. Standard prediction techniques frequently struggle to handle complex temporal dependencies in the presence of high data variability and possible anomalies. RNNs are the best tool for analysing and forecasting transaction flows in this situation, particularly their LSTM variant.

The ability of LSTM to retain information about long-term dependencies is one of its key benefits, this is particularly crucial when examining transaction sequences. Prior to training the model, thorough preprocessing is done, including data normalisation, outlier removal, and time series ordering, because the quality of the input data has a significant impact on prediction accuracy. By removing inconsistencies and guaranteeing high transaction sequence reliability, the preprocessing stage of the preprocessing process can greatly enhance the structure of the input data by utilising the PBFT and RAFT consensus algorithms.

After preparing the data, the LSTM model is trained as follows:

Model Architecture. An LSTM model for transaction prediction consists of several components, each of which performs specific mathematical operations on the data [11,12]:

- Input Layer: The input layer takes in normalized time sequences of data, which represent the transaction time series. Let the input sequence x_t have length T (the total number of time steps) and N (the number of features at each time step), then the input data can be written as a matrix:

$$X = \{x_1, x_2, \ldots, x_T\} \tag{1}$$

  where $x_t \in \mathbb{R}^N$
- LSTM Layers: The main computation in LSTM is to update the hidden states h_t and the cell state c_t at each time step t. LSTM uses several operations to process input data and store information in long-term and short-term memory:
  1. Input gate i_t:

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

  2. Forget gate f_t:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3}$$

3. Output gate o_t:
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{4}$$
4. Candidate for new cells $\tilde{c}_t$:
$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{5}$$
5. Update state of cell c_t:
$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \tag{6}$$
6. Update hidden state h_t:
$$h_t = o_t \cdot \tanh(c_t) \tag{7}$$

In these formulas:
- W_i, W_f, W_o, W_c are weight matrices for input, forget, and output gates, as well as for cell candidates.
- b_i, b_f, b_o, b_c are biases.
- σ is the sigmoid, tanh is the hyperbolic tangent.

– Output Layer: After processing all the input time steps, LSTM passes the result to the output layer, which is a simple linear regressor to predict future transactions:
$$\hat{y_t} = W_y \cdot h_t + b_y \tag{8}$$
where $\hat{y_t}$ is the future transaction prediction, W_y and by b_y are the output parameters.

Optimizing Hyperparameters. To achieve the best results, it is necessary to optimize the hyperparameters of the model:

1. Window Size k: The window size (the number of previous time steps to predict the next one) is determined by the value kk. The window determines the number of inputs that will be fed to the LSTM at each training step.The window size can be defined as:
$$X_{\text{window}} = \{x_{t-k}, x_{t-k+1}, \ldots, x_{t-1}\} \tag{9}$$
where k is the window length.
2. Number of LSTM layers: The number of LSTM layers L controls the depth of the network. Typically 2–3 layers are used, which allows extracting more complex relationships from time series.
3. Activation function:
   – LSTM uses ReLU (Rectified Linear Unit) or tanh for activation.
   – Linear activation is usually used for the output layer.
4. Optimizer: Adam or RMSprop algorithms are often used to optimize the model, which effectively adapt the learning step at each step:
$$\theta_t = \theta_{t-1} - \eta \cdot \frac{m_t}{\sqrt{v_t} + \epsilon} \tag{10}$$
where η is the learning rate, m_t and v_t are the first and second order moments, ϵ is a small value for numerical stability.

Training Process. At each training step, the input data X_{window} is fed to the LSTM, which updates the hidden state h_t and the memory cell c_t based on the previous data. After the model makes a prediction $\hat{y}_t$, the error is calculated:

$$\text{Error} = \hat{y}_t - y_t \tag{11}$$

where y_t is the true value of the transaction. The error is propagated back through the network to adjust the weights using the gradient descent algorithm.The weight matrices W and bias b are updated using gradient descent:

$$W_i = W_i - \eta \cdot \frac{\partial \text{Loss}}{\partial W_i} \tag{12}$$

where η is the learning rate and $\frac{\partial \text{Loss}}{\partial W_i}$ is the gradient of the error with respect to the weights.

This process continues throughout the training epochs until the error becomes minimal and the model begins to make more accurate transaction predictions.

After Training, the accuracy of the model is assessed using error metrics:

– MSE (Mean Squared Error):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{13}$$

The lower the MSE, the more accurate the model's predictions.

– RMSE (Root Mean Squared Error)—the root of the root mean square error:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{14}$$

RMSE is used to interpret the error in the same units as the original data.

4 Experimental Results of the Work

A number of tests were carried out to assess the suggested method's efficacy. When compared to conventional data processing, the prediction error rate was decreased by using PBFT and RAFT for data preprocessing. According to experimental results, by removing noise and organising transactions, combining these algorithms with LSTM greatly increases prediction accuracy.

4.1 Data Preprocessing

PBFT and RAFT were used to process the data before it was fed into the LSTM model. These algorithms offer:

- PBFT, which removes inaccurate data resulting from malicious or chance errors.
- RAFT lowers the degree of chaos in the input data by organising and coordinating the transaction sequence. LSTM Model Architecture after preprocessing, the data was used to train a multilayer LSTM model with the following parameters:
  - Input layer: normalized transaction time series.
  - Hidden layers: two LSTM layers with 128 and 64 neurons.
  - Output layer: regression for time series prediction.

Model errors were decreased by using PBFT and RAFT for data preprocessing because they decreased noise and improved transaction consistency. The decrease in MSE and RMSE when compared to conventional techniques verified more precise predictions and improved LSTM time series adaptation. By avoiding abrupt changes in values, better data structuring guaranteed the forecasts' stability. Therefore, combining PBFT and RAFT with LSTM improved the model's resilience and the precision of transaction flow forecasting (Table 3).

4.2 Reducing the Error Rate

When comparing data processing methods, the following improvements in model accuracy are observed:

Table 3. Accuracy of the observation model

Data processing method	RMSE	MAE	MAPE
Normal processing (without PBFT and RAFT)	0.082	0.065	8.5%
PBFT + RAFT	**0.047**	**0.032**	**4.1%**

Where:

- RMSE (Root Mean Square Error) – root mean square error;
- MAE (Mean Absolute Error) – mean absolute error;
- MAPE (Mean Absolute Percentage Error) – mean absolute percentage error.

4.3 Impact on Noise Immunity

The model's sensitivity to anomalies was demonstrated by the 15–20% error spikes it experienced in the absence of PBFT and RAFT. Preprocessing was used to stabilise the predicted values and reduce noise.

During the model training process, the Train Loss and Validation Loss graph is displayed in Fig. 3. The number of epochs is displayed on the X axis, and the loss function value is displayed on the Y axis.



Fig. 3. The process of training the model using P-BFT and RAFT transactions

1. Training Loss - measures how well the model predicts the data it is trained on. Validation
2. Loss - shows how well the model generalizes knowledge to new data.

The graph demonstrates that:

- Effective learning starts when the error rapidly drops in the initial epochs.
- Following the fifth epoch, the decline's dynamics slow down before a mild oscillation starts.
- The fact that the validation error is still less than the training error could be a sign of regularisation techniques being used or of the test data being simple.

Experimental results showed that integrating PBFT and RAFT before processing in the LSTM model allows: Reducing prediction errors (up to 50%), Increasing resistance to noise, Streamlining the transaction sequence

5 Conclusion

When combined with LSTM, the consensus algorithms PBFT and RAFT show great promise for transaction prediction in distributed systems. This method can greatly improve the consistency and dependability of data, which lowers the number of errors and raises prediction accuracy.

By combining PBFT and RAFT, anomalies from potentially malicious or random data failures are removed, and time series are smoothed, increasing the model's resistance to noise and sudden changes in input data. This is particularly crucial in blockchain and financial systems, where prediction accuracy is essential to avoiding mistakes, fraud, and poor choices.

Additionally, temporal dependencies in transaction sequences can be considered when processing ordered and cleaned data using LSTM, which reduces information loss and produces more accurate forecasts. Thus, the model's stability and adaptability to dynamic changes in distributed environments are greatly enhanced by the combination of recurrent neural networks and consensus algorithms.

References

1. Bogdanov, A., et al.: Testing and comparative analysis of the F-BFT-based DLT solution. In: Gervasi, O., et al. (eds.) ICCSA 2021. LNCS, vol. 12952, pp. 31–41. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86973-1_3
2. Bogdanov, A., et al.: Integration of PBFT and raft algorithms with recurrent neural networks to improve the reliability of distributed systems. In: Gervasi, O., Murgante, B., Garau, C., Taniar, D., Rocha, A.M.A.C., Faginas Lago, M.N. (eds.) ICCSA 2024, pp. 226–237. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-65154-0_14
3. Castro, M., Liskov, B.: Practical byzantine fault tolerance. In: OsDI, vol. 99, pp. 173–186 (1999)
4. Ongaro, D., Ousterhout, J.: In search of an understandable consensus algorithm. In: 2014 USENIX Annual Technical Conference (USENIX ATC 14), pp. 305–319 (2014)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
6. Goodfellow, I., et al.: Deep Learning, vol. 1, no. 2. Cambridge, MIT Press (2016)
7. Li, X., Liu, Q., Wu, Y.: Prediction on blockchain virtual currency transaction under long short-term memory model and deep belief network. Appl. Soft Comput. **116**, 108349 (2022)
8. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
9. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015)
10. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
11. Brown, T., et al.: Language models are few-shot learners. Adv. Neural Inf. Process. Syst. **33**, 1877–1901 (2020)
12. Wang, B., et al.: Exploring anomaly detection and risk assessment in financial markets using deep neural networks. Int. J. Innov. Res. Comput. Sci. Technol. **12**(4) (2024)