Electronic Research Archive

http://www.aimspress.com/journal/era

ERA, 31(10): 6525–6551.
DOI: 10.3934/era.2023330
Received: 09 July 2023
Revised: 21 September 2023
Accepted: 26 September 2023
Published: 11 October 2023

Research article

F-LSTM: Federated learning-based LSTM framework for cryptocurrency price prediction

Nihar Patel[1], Nakul Vasani[1], Nilesh Kumar Jadav[1], Rajesh Gupta[1,*], Sudeep Tanwar[1,*], Zdzislaw Polkowski[2], Fayez Alqahtani[3] and Amr Gafar[4]

[1] Department of CSE, Institute of Technology, Nirma University, Ahmedabad, India

[2] Department of Humanities and Social Sciences, The Karkonosze University of Applied Sciences in Jelenia GÅLora, Poland

[3] Software Engineering Department, College of Computer and Information Sciences, King Saud University, Riyadh 11437, Saudi Arabia

[4] Mathematics and Computer Science Department, Faculty of Science, Menofia University, Shebin Elkom, Egypt

* **Correspondence:** Email: sudeep.tanwar@nirmauni.ac.in, rajesh.gupta@nirmauni.ac.in.

Abstract: In this paper, a distributed machine-learning strategy, i.e., federated learning (FL), is used to enable the artificial intelligence (AI) model to be trained on dispersed data sources. The paper is specifically meant to forecast cryptocurrency prices, where a long short-term memory (LSTM)-based FL network is used. The proposed framework, i.e., *F-LSTM* utilizes FL, due to which different devices are trained on distributed databases that protect the user privacy. Sensitive data is protected by staying private and secure by sharing only model parameters (weights) with the central server. To assess the effectiveness of *F-LSTM*, we ran different empirical simulations. Our findings demonstrate that *F-LSTM* outperforms conventional approaches and machine learning techniques by achieving a loss minimal of 2.3×10^{-4}. Furthermore, the *F-LSTM* uses substantially less memory and roughly half the CPU compared to a solely centralized approach. In comparison to a centralized model, the *F-LSTM* requires significantly less time for training and computing. The use of both FL and LSTM networks is responsible for the higher performance of our suggested model (*F-LSTM*). In terms of data privacy and accuracy, *F-LSTM* addresses the shortcomings of conventional approaches and machine learning models, and it has the potential to transform the field of cryptocurrency price prediction.

Keywords: federated learning; LSTM; decentralization; cryptocurrencies; weight sharing

1. Introduction

The nation's economy is essential to contemporary society that impacts people's lives in numerous ways, such as employability, goods and service costs. The financial market is a crucial part of the economy, where investors purchase and sell different financial instruments, including stocks, bonds and cryptocurrencies [1]. Cryptocurrencies have become a popular investment choice for traders and investors due to their astounding volatility and the possibility of making big returns. Cryptocurrencies like Bitcoin and Ethereum have seen substantial price swings due to their decentralized nature and constrained supply, providing chances for traders to profit from market moves. Many people have been drawn to learn more about the world of cryptocurrencies as a way to diversify their investment portfolios due to the appeal of large returns in a relatively short amount of time.

The financial market has various stakeholders, including traders, investors, regulators and financial institutions. The cryptocurrency market is extremely dynamic, where market sentiments, news and events are few variables that can affect cryptocurrency's price. Since, the cryptocurrency prices are so erratic, it is difficult to predict it with some degree of certainty [2]. As a result, traders and investors frequently use methods including technical analysis, fundamental research and market trends to forecast cryptocurrency prices. However, technical analysis and market trends are the foundation of conventional methods for forecasting cryptocurrency values [3, 4]. To forecast the future trends of cryptocurrency prices, the technical analysis examines historical market data, i.e., price and volume. This strategy makes the assumption that previous market data can provide insight into potential market developments. Traditional approaches have several shortcomings that reduce their accuracy, and for instance, technical analysis does not take into consideration fundamental variables like news, events and market sentiment that can impact cryptocurrency prices [5]. Similarly, market patterns can unexpectedly shift, making predicting prices difficult. Thus, conventional approaches could not yield accurate predictions, particularly in extremely volatile markets like the one for cryptocurrency.

Machine learning techniques can be used to increase the precision of price predictions for cryptocurrencies [6–9]. Machine learning models analyze the historical market data (price and volume) to forecast future trends and identify intricate data patterns that conventional approaches could miss. Machine learning models do, however, have several drawbacks, such as overfitting and data bias. The accuracy of predictions may be impacted by these restrictions, particularly if the training data is sparse or biased. Deep learning techniques, particularly the LSTM model, have demonstrated promise in predicting cryptocurrencies' prices to overcome them [10, 11]. Traditional machine learning models might be unable to capture the temporal dependencies in the data that the LSTM model can do. LSTM model can also handle the high-dimensional, non-linear data that characterizes financial markets. However, since LSTM models need a lot of training data, centralized systems might not have it.

The aforementioned issues can be resolved using federated learning. It is also a remedy for the issue of data privacy in centralized systems. Model training on decentralized data sources is made possible using FL without compromising the data privacy [12]. In this, the model is trained using local data sources, and only the model parameters (weights) (not the raw data) are shared. In order to increase accuracy while maintaining data privacy, we present a federated learning-based LSTM model for cryptocurrency price prediction in this study. The suggested approach addresses the shortcomings of conventional and machine learning techniques while utilizing the benefits of LSTM models. With

the LSTM models being trained locally on remote devices with incremental projections of data over time, the models increase the accuracy and precision in local devices, and through the utilization of a FL framework, the global model adapts to the incrementally learned weights. The proposed framework provides a solution to higher optimization and privacy preservation. The modular framework overall thus achieves a significant decrease in loss with time using simple $FedAvg$, making the system efficient, optimized, privacy preserved and secured. The loss after global iterative training is at the minimum of 2.3×10^{-4} without any data transfer to a centralized server. Furthermore, the proposed model operates at an exceptionally low memory and CPU cost and gives highly accurate predictions.

1.1. Research contributions

The following are the contributions of this paper.

- We proposed a unique Federated Learning-based LSTM model for cryptocurrency price prediction that improves prediction accuracy and gives an almost negligible loss compared to traditional and machine learning methods.
- A significant outcome of this paper on FL is the ability to train models on decentralized data sources without compromising data privacy. The central server receives no raw data from the proposed model, only its model parameters (weights). Increased model openness, accountability and audibility are also made possible by the decentralized approach.
- Another significant contribution is that the suggested model can be expanded while retaining a high level of accuracy to accommodate a large number of participants. The model may be trained on many devices at once thanks to federated learning, which enables model training on decentralized data sources. This strategy can shorten the training period, increase the model's accuracy and diversify the training data.
- The Proposed framework Federated Integrated LSTM model *F-LSTM* presents a highly efficient system for forecasting prices without actually training on the original set of sequential data. The Framework works in a dynamic environment for both discrete and sequential data forms. Through the realization of the incremental learning technique, the model works in real-time for continuously evolving data with a constant and steady loss of 2.3×10^{-4}.

1.2. Organization

The arrangement of the paper is as follows. Section 2 defines the novelty in our proposed framework. The relevant works in the area of price prediction are described in Section 3. Section 4 comprises the system Model and problem formulation for the same. The proposed *F-LSTM* framework for cryptocurrency price predictions is presented in Section 5. The focus of Section 6 is on the outcome and analysis of the suggested framework. The paper is concluded in Section 7.

2. Novelty

In this study, we present a novel sequential framework that deviates from the usual predictive abilities connected with conventional LSTM models. We are not primarily concerned with making predictions as is done in the past as presented in [13] and [14]. Instead, we suggest a novel strategy

designed to give consumers a computationally effective substitute as described in [12], particularly for local device training. Our system is specifically made to drastically cut down on overall computing costs, outperforming current state-of-the-art models in this regard.

Our framework's ability to strike a careful balance between maintaining the predictive power of individual models and streamlining the training of a global model is one of its fundamental advances. Notably, this optimization is accomplished without requiring repeated data access. In summary, while we effectively adjust the weights of the global model, our architecture ensures that the predictions made by individual models remain intact. This is done without having to access often and manipulate huge amounts of data or keep up a centralized data repository.

The proposed framework is meticulously built to comply with the highest optimization requirements, producing a worldwide model with remarkable precision as a result. Our method is unique because it can do rid with the necessity for centralized data storage and comprehensive data access. With the help of this special feature, distributed sequential learning will be able to explore new and intriguing directions without having to make significant structural changes.

3. State-of-the-art works

This section discusses the related work carried out by researchers in the field of stock and cryptocurrency price prediction. Numerous studies have explored different methodologies and approaches to forecast prices, aiming to provide valuable insights for decision-making, risk management and investment strategies. Traditional price prediction methods use machine learning and deep learning methodologies to predict prices based on historical data. For instance, reference [15] presented a transformer-based attention mechanism for stock price prediction. The attention mechanism helps the model to focus on the most significant portions of the input sequence when generating the output sequence. This results in the extraction of the most important features. However, training such a framework requires huge computational resources and high training time.

Further, references [16] and [17] depicted frameworks developed on hybrid models, combining multiple machine and deep learning techniques. The authors in [16] proposed a FinBERT-LSTM deep learning-based stock price prediction model with news sentiment analysis. It gives an accuracy of 0.9859 for the NASDAQ-100 index stock while giving a remarkably low mean absolute percentage error of 0.014. However, this framework is overly dependent on the quality of the FinBERT model and requires extensive parameter tuning for optimal results on varied datasets. On the other hand, reference [17] proposed a hybrid model that encapsulates attention-based convolutional neural network (CNN)-LSTM and XGBoost for stock price prediction. The hybrid nature of the model improved robustness and generalizability. However, the working of their framework requires high computing power, and the combination of the attention mechanism and XGBoost restricts the interpretability of the framework to a certain extent.

Later, reference [12] presented a federated learning-enabled predictive analysis model to forecast stock market trends. Federated learning-based random forest and support vector machine (SVM) models are used in its development. Random forest gave a minimum squared error of 0.021, while the SVM model, when deployed, gave a minimum squared error of 37.596. FL permits privacy-preserving and distributed learning. This led to its ability to leverage diverse data sources. However, using random forest and SVM leads to limited accuracy and a high error score. Better

algorithms like LSTM or Bi-directional LSTM can boost the prediction accuracy and minimize the loss. The authors of [18] presented a Bi-LSTM network to predict the price of Bitcoin. It is a recurrent neural network consisting of two LSTM layers processing the input sequence in both forward and backward directions to capture long-term dependencies in sequential data. The Bi-LSTM Network captures dependencies in both forward and backward directions, resulting in a good RMSE. However, it is likely to suffer from overfitting and cannot handle anomalies in the market [19].

Furthermore, reference [20] proposed an integrated approach using hybrid LSTM-ELM for Bitcoin price forecasting. LSTM and ELM algorithms are combined to create a hybrid LSTM-ELM model (Long Short-Term Memory - Extreme Learning Machine) to increase the precision of time-series predictions. The model's LSTM layer captures the temporal dependencies in the input sequence, and the ELM layer is employed as a quick and effective way to train the network's output layer. This results in high accuracy (0.9397 and 0.9469 in two low volatility periods and 0.9328, 0.9004 and 0.9525 in three high volatility periods), along with minimal error due to the high effectiveness of the hybrid model. However, this model is likely to suffer from overfitting due to its complexity and requires a large amount of historical data to be trained effectively. After this, reference [21] presented a CNN-LSTM Model for cryptocurrency price forecasting. The dual algorithm nature of this model allows it to capture both short-term and long-term dependencies in the input data to maintain high accuracy and precision. Another advantage of this model is that it can handle sequential and non-sequential data. Additionally, the framework has been evaluated on a variety of evaluation metrics. However, it proves to be very computationally expensive, and the effectiveness of predictions may differ for varying cryptocurrencies. Finally, reference [22] gives an LSTM-based Bitcoin price prediction framework. An optimized LSTM is used, which gives a phenomenally low error of 288.5989. Another merit of using LSTM is that it can learn to forget or retain information selectively. However, the naïve LSTM model is not generalizable for other cryptocurrencies and improved accuracy and lower error can be achieved if combined with other techniques.

There are some methods that can be used to predict stock market prices, which are described by [13]. This study of deep learning frameworks concisely describes several deep learning frameworks that can be used to predict stock market trends. Some of them are Large language models (LLM), Time series forecasting, deep neural networks and feature engineering for identifying relevant features. Reference [14] reviews a wide variety of machine learning and deep learning techniques for predicting the stock market prices by utilizing the power of fundamental indicators such as PE ratio and moving averages, which are effective in the long run but come with the factor of uncertainty to forecast the price action with minimum loss.

In the relevant works mentioned above, different methods for price prediction employing machine learning techniques were highlighted. However, these methods have inherent drawbacks and are frequently insufficient for various reasons, including high computational demands, a lack of privacy and centralization difficulties. We provide an LSTM-based decentralized, federated learning strategy that eliminates several drawbacks. Our suggested model uses fewer computational resources and achieves a smaller loss in time-series prediction tasks, making it a more effective and privacy-preserving option. Overall, our suggested model overcomes many of the shortcomings of previous approaches and provides a promising solution for time-series prediction applications. Table 1 displays the relative comparison between the state-of-the-art works and the proposed work.

Table 1. Comparative analysis of the proposed framework with the existing state-of-the-art schemes for cryptocurrency price prediction.

Authors	Year	Description	Algorithms Used	Merits	Demerits
Ardakani et al. [12]	2023	Uses SVM as a base model for identifying and recognizing patterns while integrating the base model with federated learning for decentralizing the learning model	SVM integrated with Federated Learning	Enables privacy-preserving; can leverage diverse data sources	Poor accuracy/loss
Jiang et al. [13]	2021	Presents several state-of-the-art models and frameworks that are proficient and widely used in forecasting sequences and prices.	ARIMA, LLM, GARCH	Develops a concise understanding of the approaches that can be used in a synergistic way to predict market prices	The study of the models is clear, but use in a combined manner is not explained
An et al. [14]	2022	Proposes various machine learning and deep learning procedures to predict stock market price actions.	Regression, Classification	A concise approach for versatile use of algorithms	No detailed description of models
Zhang et al. [15]	2022	Transformers used for learning the sequences of the prices in the financial market data and accurately regenerate the useful features and patterns in the sequences.	Transformer Encoder-based Attention Network (TEANet) framework	Effective feature capturing	Computationally expensive; high training time
Halder [16]	2022	Deep LSTM model integrated with pretrained language model with the add-on of sentiment analysis for reducing the randomness and capable of capturing nuanced languages and context specifications.	FinBERT-LSTM	Good accuracy (0.9859); two technologies	Overly dependence on FinBERT; extensive parameter tuning
Shi et al. [17]	2021	Combines CNN-LSTM and XGBoost by catching important portions of time series data using an attention-based mechanism by integrating a hybrid approach of CNN learning and sequence learning patterns.	CNN-LSTM and XGBoost hybrid model	Robust; generalizable; able to incorporate multiple sources of data	Computationally expensive; limited model interpretability
Nithyakani et al. [18]	2021	Uses Bi-LSTM model to capture both past and future dependencies in the sequential data for predicting the prices if Bitcoin	Bi-LSTM	Captures dependencies in both forward and backward directions(past and future)	Overfitting; flawed anomaly handling
Luo et al. [20]	2022	Integrates LSTM with ELM machine learning models for predictions and forecasting of input sequences where the ELM algorithms enhance the predictions made by LSTM	Hybrid LSTM-ELM	high accuracy (0.9512); effective model	Overfitting; requires a large amount of data for effective training
Livieris et al. [21]	2021	Uses image representation of historical prices of cryptocurrency for forecasting future prices by integrating CNN with LSTM using a sliding window approach by utilizing both spatial and temporal features	CNN and LSTM	minimal error (MAE is 0.005 and RMSE is 0.007) & Precision; handles both sequential and non-sequential data	Computationally expensive; tentative overfitting
Ferdiansyah et al. [22]	2019	Uses normal LSTM model for forecasting the Bitcoin prices on the sequential data of Yahoo Finance	LSTM	Very low error (RMSE is 288.59); can selectively forget or retain information	Generalizability issues
Proposed framework	2023	*F-LSTM*: Federated Learning-based LSTM Framework integrated with Incremental learning on the sequential data for cryptocurrency price prediction	Federated Learning-based LSTM framework	high precision; minimal error(MSE is 0.0002); decentralization; enhanced privacy and security; high generalizability of model	-

4. System model and problem formulation

This section presents the system model and problem formulation for the proposed *F-LSTM* model. In the everlasting global economy of randomness and rapid changes, the most variably-correlated factor for forecasting the trends of the current economy is the stock market trends. These trends efficiently describe the economic situation of a country or a group of organizations by weighted aggregation of independent factors regressed together. Many fellow researchers made great attempts to forecast the trends of stock markets and have been successful to attain the closest accuracy possible through the use of a heavily trained transformers model. This accuracy is attained at a heavy cost of the vast utilization of computational resources embedded together for the training over the large dataset. The proposed architecture aims to present an optimized solution by solving both the

optimization and scalability problems of the previously trained models. In the proposed system, the client-server is established as individual devices $\{C_1, ..., C_i, ..., C_j, ..., C_m\} \in C$ where the client is referred to as a diversified entity which can be described as an individual or an organization with individual data consisting of respective stock market fluctuations upon a myriad number of stocks. The set $\{T_1, ..., T_i, ..., T_j, ..., T_m\} \in T$ represents the trends or fluctuations in the value of these stocks. These trends are monitored by LSTM models, $\{L_1, ..., L_i, ..., L_j, ..., L_m\} \in L$, which are deployed by different investors or financial institutions.

Equations (4.1) and (4.2) represent a stock that can exhibit one or more trends.

$$\exists C_i \in C \xrightarrow{has} T_i \in T \Leftrightarrow C_i, T_i \tag{4.1}$$

$$\exists C_i \xrightarrow{has} T_2, T_3, ..., T_j \in T \tag{4.2}$$

In Eqs (4.3) and (4.4), if a cryptocurrency exhibits a trend, it is categorized as an "active cryptocurrency"; otherwise, it is a "dormant cryptocurrency".

$$C_i \cap T_i \neq \emptyset \rightarrow ActiveClient \tag{4.3}$$

$$C_i \cap T_i = \emptyset \rightarrow DormantClient \tag{4.4}$$

Upon successful categorization of a cryptocurrency trend's occurrence on a client-server, respective clients are characterized as active or dormant. The LSTM model learns from the historical data and makes forecasts upon the training from the input stock data, and the global FL model identifies the active client weights and dormant client weights upon applying the *Federated Averaging* algorithm to each client server transition.

The system uses FL to solve the problem of scalability. With federated learning, each LSTM model at the edge node (investor or financial institution) is trained on its local data. The models then send their learnings (model's updated weights) to a central server, where they are aggregated to form a global model weight. These global model weights are then sent back to the edge nodes to reset the individually trained weights of the nodes, and the process iteratively optimizes the proposed approach. Furthermore, this approach allows for training on a much larger dataset than what a single LSTM model could handle, leading to more accurate and robust predictions.

The optimization problem is addressed by tuning the LSTM models and the FL process to maximize the accuracy of the predictions while minimizing the computational and communication costs. This involves optimizing the number of layers in the LSTM models, the number of iterations in the FL process and the updated model weights.

5. The proposed framework

This section describes the proposed system model architecture Ψ (as shown in Figure 1) for forecasting the prices of a particular cryptocurrency, Ethereum, upon training on the determined span of years. The framework is divided into 3 distinct layers, i.e., data, AI and application layers. The functionality of each layer is described as follows.

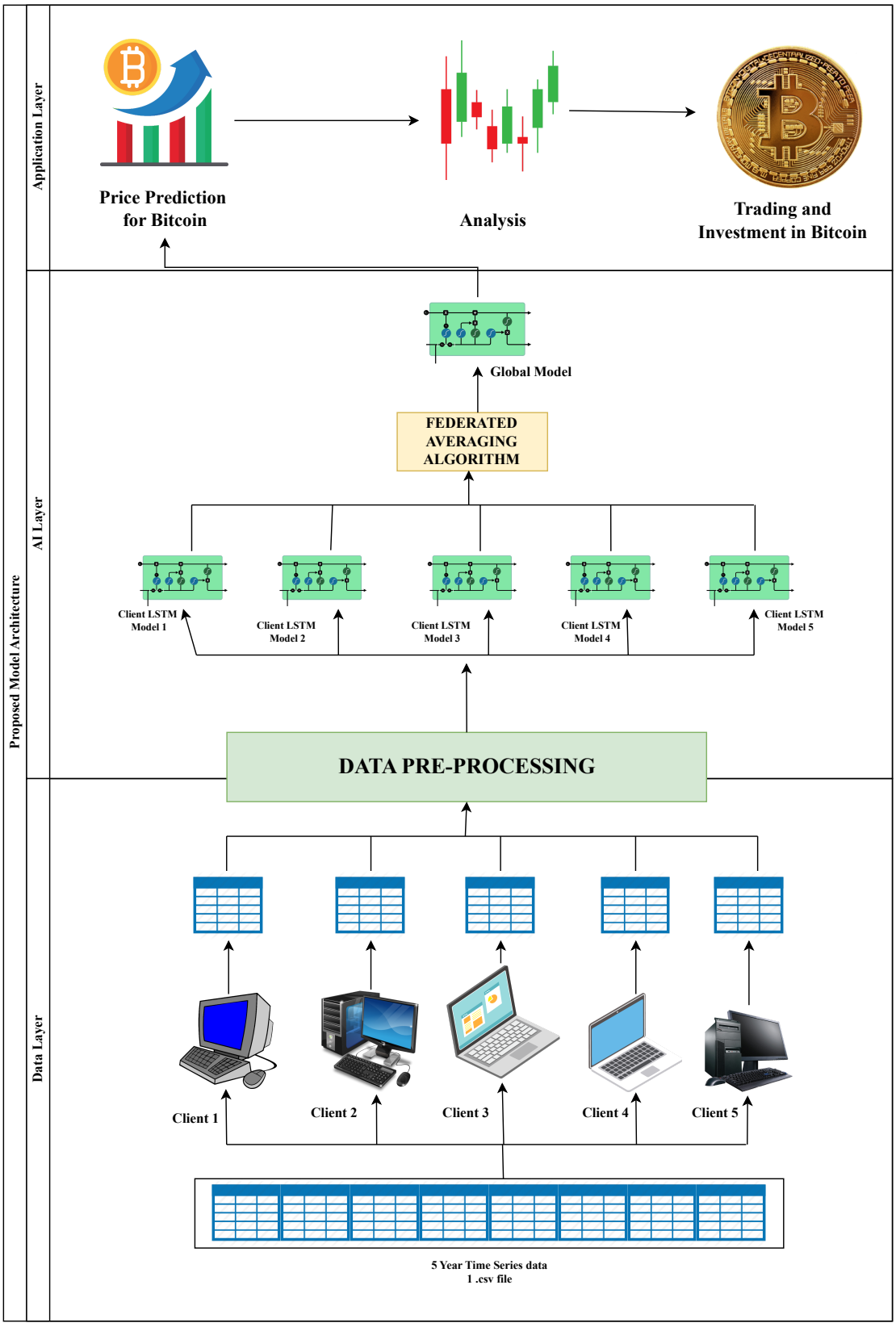


Figure 1. System architecture.

5.1. Data layer

The data layer includes several financial markets $\{M_1, ..., M_i, ..., M_j, ..., M_m\} \in M$ that can be used to track market activity and record stock price fluctuations. These markets distribute real-time stock prices, which the suggested framework (LSTM model) then uses to forecast the future occurrences of trends. From the price data, it is to determine whether any significant price movement for the stocks $\{S_1, ..., S_i, ..., S_j, ..., S_m\} \in S$ will occur and classify them as bullish (price likely to rise) or bearish (price likely to fall). This is determined by whether or not the stock exhibits any specific trend $\{T_1, ..., T_i, ..., T_j, ..., T_m\} \in T$ in the price data. The data is captured in the form of a time series and is further forwarded to the AI layer, which makes overall predictions. This is done through the FL process, where the LSTM model is trained across multiple decentralized edge devices, each corresponding to a particular financial market. The model learns from the local data on each device, and the global model is updated by aggregating these local updates by the data available on the local devices.

5.2. Artificial intelligence layer

The AI layer consists of multiple LSTM models $\{L_1, ..., L_i, ..., L_j, ..., L_m\} \in L$, each corresponding to a specific financial market within the set $\{M_1, ..., M_i, ..., M_j, ..., M_m\} \in M$. These LSTM models are utilized to predict stock price trends and movements for each stock $\{S_1, ..., S_i, ..., S_j, ..., S_m\} \in S$ in the respective financial market. The stocks are distributed among the set of clients $\{C_1, ..., C_i, ..., C_j, ..., C_m\} \in C$. Each client has LSTM models $L_i \in L$ integrated within the system. These models are trained on a particular stock $S_i \in S$ tuned to identify the trend $T_i \in T$ in the data. Further the respective weights $\{W_1, ..., W_i, ..., W_j, ..., W_m\} \in W$ of each model are sent to the global server for aggregation and sent back to a cluster of distributed clients. In a bullish trend, the stock price is expected to rise, while in a bearish trend, it is expected to fall. Equations (5.1) and (5.2) represent the LSTM model's capability to classify a stock's trend based on the time-series data.

$$LSTM(L_i) \rightarrow S_i \cap T_i \neq \emptyset \rightarrow Bullish \tag{5.1}$$

$$LSTM(L_i) \rightarrow S_i \cap T_i = \emptyset \rightarrow Bearish \tag{5.2}$$

The LSTM models are trained using federated learning. The learning process takes place across a cluster of decentralized edge devices, each corresponding to a specific financial market. Each LSTM model learns from the local time-series data, after which the weight updates are aggregated to form the global model weights. These global model weights are then sent back to the local LSTM cluster edges, leading to an improved second round of learning. This process continues in iterative cycles, leading to progressively more accurate predictions over time. The AI Layer thus not only performs the predictive analysis but also ensures the preservation of privacy along with scaling the model over distributed systems and optimizing the performance throughout each iteration by the use of FL systems. The FL approach enables the AI layer to handle large-scale data from multiple financial markets without the need for centralization, making it a powerful tool for predicting stock market trends.

5.2.1. Dataset description

The dataset used to train the *F-LSTM* model is obtained from the *Yahoo Finance API*, which provides large-scale historical data of many organizations and global market trends [23]. It

conventionally comprises of 7 attributes *Open*, *High*, *Low*, *Close*, *Volume*, *Dividend*, *Stock Split*. The time frame length taken is 1825 time series points for model training. In the proposed system, numerous trends and fluctuations occur over a period of time $\{T_1, ..., T_i, ..., T_j, ..., T_m\} \in T$ and these trends are recognized by the formulated model.

5.2.2. Dataset preprocessing

First, we store the multivariate time-series data δ in cloud storage and subsequently load it for preprocessing to enhance training accuracy and prediction outcomes. Initially, the data is normalized to a standard range, typically between 0 and 1, to manage outliers and ensure the model can handle extreme values effectively. The mathematical representation of normalization is described as follows.

$$\delta \rightarrow (x, y) \rightarrow (\frac{x}{x_{max}}, y) \tag{5.3}$$

Equation (5.3) represents the mapping and scaling of data, where x is the variable used to denote the tensors comprising values of the multivariate time-series data, and y is the variable used to denote the label or target variable. The data is configured in a way such that a multivariate base LSTM model can be designed to make accurate and precise predictions. This is done by parsing the data comprising attributes *Open*, *High*, *Low*, *Close* and *Volume* into a multi-dimensional *Numpy* array with all the respective attributes taken into consideration.

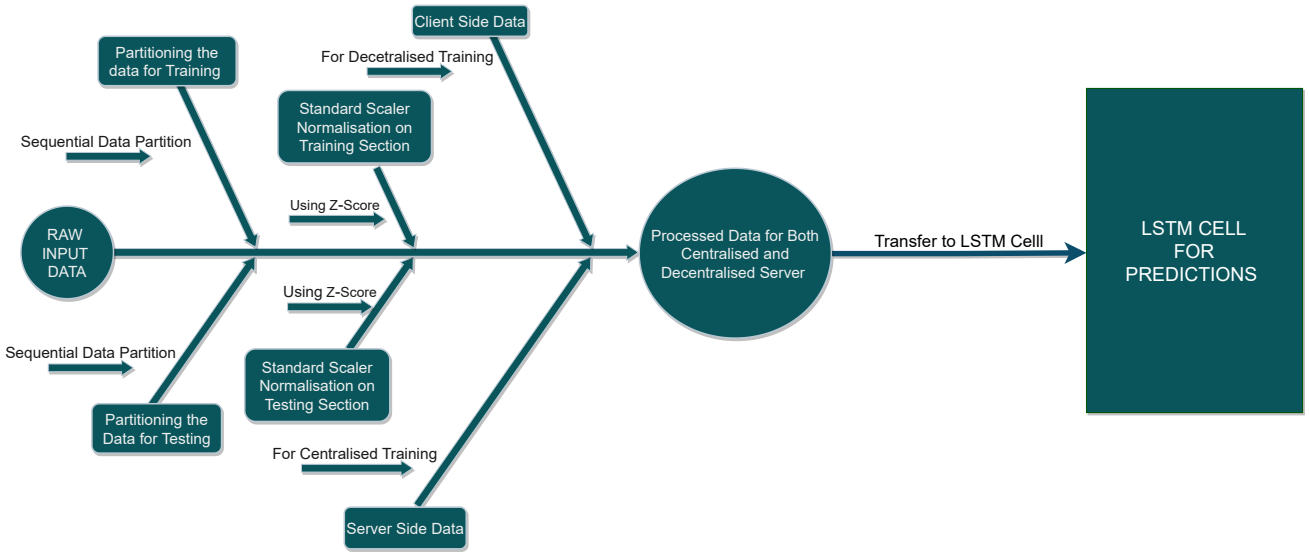


Figure 2. Data preprocessing.

The *Numpy* array is then scaled and normalized for better performance and optimized computation while training. The scaled data is divided into a training set (80% of the data) and a test set (20% of the data). It is also ensured that the test data includes a certain number of previous time steps (defined by sequence length, set to 50) before the start of the test period. In the proposed model, FL is integrated, which means that each LSTM model learns from its local batch of data. After local training,

the models' updates are sent to a central server where aggregation for computing the global weights is performed. These calculated global weights are then sent back to each local LSTM model, enhancing the subsequent learning cycle. This iterative process progressively refines the model's predictions over successive cycles. The entire flow of data preprocessing is shown in Figure 2.

5.2.3. Model motivation

The inability of conventional and machine learning methods to effectively predict the extremely volatile and non-linear cryptocurrency market is the primary motivation behind the development of a *F-LSTM* model for price prediction. The traditional approaches are based on historical data and stationary state assumptions, which are not always valid in the complicated and volatile cryptocurrency market. Similarly, because machine learning models are trained using centralized data sources that might not accurately reflect the variety of the market, they are susceptible to bias and overfitting. Furthermore, privacy issues are a crucial problem when predicting cryptocurrency prices because centralized data sources may put sensitive information at risk. Therefore, a decentralized and safe method of cryptocurrency price prediction is required, one that guarantees high accuracy while protecting the privacy of sensitive data. Furthermore, privacy issues are a crucial problem when predicting cryptocurrency prices because centralized data sources may put sensitive information at risk. Therefore, a decentralized and safe method of cryptocurrency price prediction is required, one that guarantees high accuracy while protecting the privacy of sensitive data.

These issues are resolved by *F-LSTM* framework, which enables the historical pattern learning algorithm in a multimodal environment. The algorithms are made intact to adapt with increments of data shards into the local client repository and learn over time. The framework thus identifies the effective sequential patterns in the data and combines local models' individual understanding to tune and optimize the global model. The global model thus learns better than individual clusters and generates an optimized solution. This solution as a whole is considered to be the versatile answer for application-based software who are destined to learn on a large chunk of data and have to evolve on the newly available data continually. Thus, considering all the constraints for satisfying evolutionary incremental learning along with preservation of the individual data characteristic *F-LSTM* framework, proves to be an efficient approach for building optimized software.

Furthermore, we analyzed [24], which was a study that used a model based on long short-term memory to forecast energy consumption in order to improve prediction performance, and [14], where an empirical study was done on two cases of pork prices and soybean futures prices, and 12 comparative prediction models were developed based on random forest (RF), LSTM and multilayer perceptron (MLP). These studies demonstrated the strengths of LSTM for price prediction and hence, we were motivated to develop the proposed framework on the LSTM algorithm due to the several reasons that are stated below.

- **Capturing long-term dependencies**: The purpose of LSTM networks is to identify long-distance dependencies in sequential data. The capacity of LSTM to capture long-term patterns is vital in cryptocurrency markets, as prices are affected by previous patterns and occurrences.
- **Handling temporal dynamics**: Prices for cryptocurrencies display complex temporal dynamics, including seasonality and volatile swings. Since LSTM is so efficient at modeling and adjusting to such temporal fluctuations, it is appropriate for situations found in dynamic markets.

- **Memory cell**: A dedicated memory cell in the LSTM can store and spread data over a long period of time. For the purpose of recalling historical pricing trends and putting them into forecasts, this feature is crucial.
- **Sequence-to-sequence learning**: Sequence-to-sequence learning is supported by LSTM, enabling it to forecast future sequences using existing price sequences as input. This is crucial for forecasting full price trajectories rather than just specific price points.
- **Robust to noisy data**: Cryptocurrency data can have outliers and sharp spikes, making it noisy. LSTM can learn to weed out irrelevant data and concentrate on the underlying patterns while being robust to noisy input.
- **Adaptability**: LSTM models are flexible and rapidly respond to shifting market circumstances. In the rapidly changing and highly volatile world of cryptocurrency, adaptation is essential.

5.2.4. Model development

The proposed *F-LSTM* model ψ is built upon a sequence-to-sequence architecture (as shown in Figure 3), with the input tensor's size determined by the training data's shape which is described in Eq (5.4).

$$n_{\text{neurons}} = x_{\text{train}.shape[1]} \times x_{\text{train}.shape[2]}. \tag{5.4}$$

The initial tensor, of size calculated from the Eq (5.4), is first passed through the LSTM block with n_{neurons} neurons, where n_{neurons} is the product of the second and third dimensions of the training data. The second dimension denotes the number of data points in the training sample, and the third dimension denotes the number of independent variables considered for the training of the ψ model.

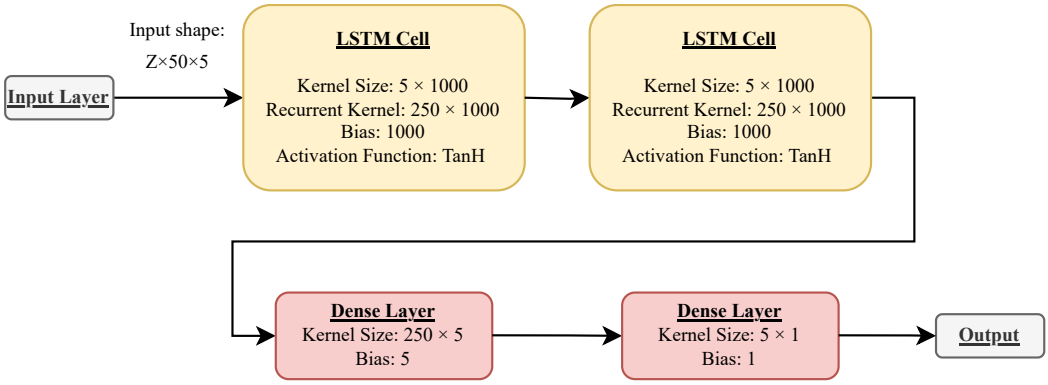


Figure 3. LSTM model architecture.

Each LSTM unit computes the current cell state c_t and the current hidden state h_t as follows:
1) *Forget gate* (f_t): Determines what information the cell state should forget.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{5.5}$$

2) *Input gate* (i_t): Determines what new information the cell state should store.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{5.6}$$

3) *Cell state ($\tilde{C}_t$)*: A candidate value for the cell state.

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{5.7}$$

4) *Update of the cell state (c_t)*: Update the old cell state c_{t-1} into the new cell state c_t.

$$c_t = f_t * c_{t-1} + i_t * \tilde{C}_t \tag{5.8}$$

5) *Output gate (o_t)*: Determines what the next hidden state should be.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5.9}$$

6) *Update of the hidden state (h_t)*: Update the old hidden state h_{t-1} into the new hidden state h_t.

$$h_t = o_t * \tanh(c_t) \tag{5.10}$$

Here, σ denotes the sigmoid function, and $*$ denotes element-wise multiplication. The weight matrices W_f, W_i, W_C and W_o, and the bias vectors b_f, b_i, b_C and b_o are iteratively updated and are relaxed to learn throughout the training of the model as an overall. In a neural network, a dense layer represents a matrix-vector multiplication. The values in the matrix are the trainable parameters (weights) that get updated during backpropagation. A conventional neural network layer l is denoted as follows.

- $x^{[l]}$ as the input to layer l
- $W^{[l]}$ as the weights of layer l
- $b^{[l]}$ as the bias of layer l
- $g^{[l]}$ as the activation function of layer l

The output after computation of the respective layer l is shown in Eq (5.11)

$$z^{[l]} = W^{[l]} \cdot x^{[l]} + b^{[l]} \tag{5.11}$$

$$h^{[l]} = g^{[l]}(z^{[l]}) \tag{5.12}$$

Following the LSTM layers, the model introduces a fully connected (Dense) layer with five neurons. This dense layer performs the weighted computation of inputs attained from the output of the previous LSTM layers. The model culminates in an output Dense layer with a single neuron. The output of this neuron is the prediction of the model. The model overall is a regression model to forecast the single next value in the sequence of the time series.

The model structure is defined, and at the later stages, it is compiled with the Adam optimizer and the evaluation metrics, i.e., the Mean Squared Error (MSE) as the loss function. The Adam optimizer is an extension to stochastic gradient descent that adapts learning rates for each weight in the model, improving the classic stochastic gradient descent. The MSE loss function is considered to be the most appropriate for regression problems and measures the average of the squares of the errors, i.e., the average squared difference between the estimated values and the actual value. To further enhance privacy preservation and data efficiency, the tuned LSTM model is augmented with an FL approach. Instead of the traditional way where data is sent to a centralized location, the FL approach allows for the model training to happen on the client devices themselves, therefore keeping the data local.

This approach is highly beneficial in scenarios where the data is sensitive or massive, which makes it challenging to centralize, and a crucial factor of privacy preservation is in need. The integration of FL with the LSTM model follows specific steps.

1) *Initialization* - A global LSTM model is initialized on the server, and its weights are shared with all the client devices.

2) *Local Training* - Each client device trains the received model on its local data. This training includes the forward pass, backward pass and weight updates, which are all performed locally.

3) *Model Aggregation* - After local training, each client sends their model updates (i.e., the changes in weights and biases) to the server. Importantly, the raw data never leaves the client's device, maintaining data privacy.

4) *Global Update* - The server aggregates the received updates from all clients and updates the global model. The aggregation can be a simple averaging or a more sophisticated method depending on the use case. The updated global model is then shared with all the clients, and the process repeats.

The crucial part of FL is the aggregation of the model updates on the server. The most common aggregation method is Federated Averaging (FedAvg), which is simply the weighted average of the model updates from all clients. Given K clients, where each client k has a weight update ΔW_k and a bias update Δb_k, the global weight and bias update ΔW and Δb can be calculated as follows.

$$\Delta W = \frac{1}{K} \sum_{k=1}^{K} \Delta W_k \tag{5.13}$$

$$\Delta b = \frac{1}{K} \sum_{k=1}^{K} \Delta b_k \tag{5.14}$$

The global model parameters (weights) are then updated using these averaged updates.

$$W = W + \Delta W \tag{5.15}$$

$$b = b + \Delta b \tag{5.16}$$

The notable factor while performing the computation in a real-time environment is the averaging can be weighted based on the number of samples each client has. Clients with more data would contribute more to the global model update. The FL process, including Federated Averaging, is done iteratively over multiple rounds until the model performance converges. This ensures that the model learns from the entire distributed data while keeping the data local to each client. The integration of FL into the LSTM model allows for efficient learning from distributed data sources while respecting the privacy of the data owners. The model training can be seen as a cooperative effort, where each client contributes to the learning process, thus enabling the model to learn from a more diverse and representative dataset.

The procedure is repeated iteratively over time to attain minimum loss from the predictions as described in the Algorithm 1.

Algorithm 1 FL-LSTM (FL with LSTM)

procedure FL-LSTM (W, b)
  Preprocess the input data using a standard scaler method
  Let X be the input data
  Compute the mean for each feature:
    $\mu = \frac{1}{N}\sum_{i=1}^{N} X_i$
  Compute the standard deviation for each feature:
    $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X_i - \mu)^2}$
  Scale the input data:
    $X_{\text{scaled}} = \frac{X-\mu}{\sigma}$
  Initialize the LSTM model with multiple LSTM layers:
    model = Sequential ()
    model.add (LSTM (units, return_sequences = True))
    ...
    model.add (LSTM (units, return_sequences = False))
    model.add (Dense (output_size))
  for each round r **do**
    Select K clients at random
    for each client k in K **do**
      Initialize $W_k, b_k \leftarrow W, b$
      Compute $\Delta W_k, \Delta b_k$ from client's preprocessed data using LSTM
      Send $\Delta W_k, \Delta b_k$ to server
    end for
    Server updates W, b as $W \leftarrow W - \sum_k \Delta W_k$, $b \leftarrow b - \sum_k \Delta b_k$
  end for
end procedure

5.3. Application layer

To provide real-time price predictions for the cryptocurrency, Ethereum, the application layer of the system architecture for the *F-LSTM* model, is essential. The model gets continuous data streams from various sources, such as exchange APIs and market data providers, at this layer, assuring the availability of up-to-date information. The LSTM model, trained using FL strategies, makes use of the combined intelligence of numerous participants while preserving the security and privacy of the data. The algorithm dynamically changes its forecasts as fresh pricing data comes in, allowing users to make wise trading and investment decisions. Once the price predictions are generated in real-time, the application layer facilitates further data analysis. This analysis can involve various techniques such as statistical analysis, pattern recognition and anomaly detection to extract meaningful insights and identify potential market trends. Traders and investors can utilize these insights to understand the market dynamics, evaluate risk factors and make informed decisions regarding their cryptocurrency portfolios. By integrating the LSTM model with the analysis capabilities at the application layer, users gain a comprehensive understanding of the cryptocurrency market, enabling them to optimize their trading strategies and potentially maximize their returns on investments in cryptocurrencies.

6. Results and discussions

In this section, we discuss the performance analysis of the proposed framework using different evaluation metrics, such as training loss, validation loss, resource utilization and computation time. A detailed result analysis is as follows.

6.1. Experimental setup and simulation parameters

The work on the proposed framework is done on a Python-based development environment, Jupyter Notebooks. Various APIs and functionalities from different libraries were used to develop the framework. The Yahoo Finance API, yfinance (), is used to get the historical and real-time data of cryptocurrencies. Two important functionalities in use to develop the framework are sklearn.preprocessing.RobustScaler () and tensorflow.keras.callbacks.EarlyStopping (), sklearn.preprocessing.RobustScaler () is used to scale features using statistics that are robust to outliers. This ensures precise standardization of the data tensorflow.keras.callbacks.EarlyStopping () is used to stop training when a monitored metric has stopped improving; this stops the wastage of computational resources. Some other libraries used were *numpy*, *pandas* and *seaborn*.

LSTM networks are employed in deep learning, a type of recurrent neural network (RNN) that can learn long-term dependencies, particularly in tasks involving sequence prediction. It has several hyperparameters that need to be set optimally for ideal results. LSTMs have a variety of applications, like price prediction, sentiment analysis, language modeling, speech recognition and video analysis. The application based on this paper is allied to price prediction for cryptocurrencies. The following sections highlight the use of the proposed *F-LSTM* model and its behavior using different performance parameters like loss, computation time, performance under different optimizers and memory utilization.

6.2. Performance analysis

6.2.1. Loss for *F-LSTM*

The loss measure refers to a statistic used to express the difference between the model's anticipated prices and the actual prices that were actually observed. The loss measure is used to gauge how well the LSTM model is doing in terms of its ability to predict outcomes. The LSTM model's performance on the training data—the dataset used to train the model—is measured by its training loss. A low training loss indicates the LSTM model is successfully picking up on and recognizing the patterns in the training set. On the other hand, An LSTM model's validation loss reflects how effectively it generalizes to data that it hasn't encountered during training. It is calculated using a distinct dataset from the training data called the validation dataset. The validation loss is a measure of the model's predicted performance on fresh or untested data. It helps in determining whether the model is capable of capturing fundamental data patterns without having to memorize the training samples. Figure 4 compares loss to validation Loss while training the model. In Figure 4, the x-axis indicates the measure of loss and the y-axis indicates the number of epochs during training. As depicted in Figure 4, loss and validation loss decrease with the increase in epochs. After robust training, the loss and validation loss converge well, which shows less over-fitting and a good model. The model, which is developed for the whole data set, is trained for a total of 50 epochs. The loss and validation loss start converging

well after 10 epochs. By the end of 50 epochs, they have almost identical values, indicating minimal over-fitting in the trained model. The layers and the number of neurons have been aptly defined before training the model, due to which the model has been sufficiently trained, leading to minimal or no over-fitting. Another reason for such a minute level of over-fitting is that the model has been trained up to the right number of epochs.

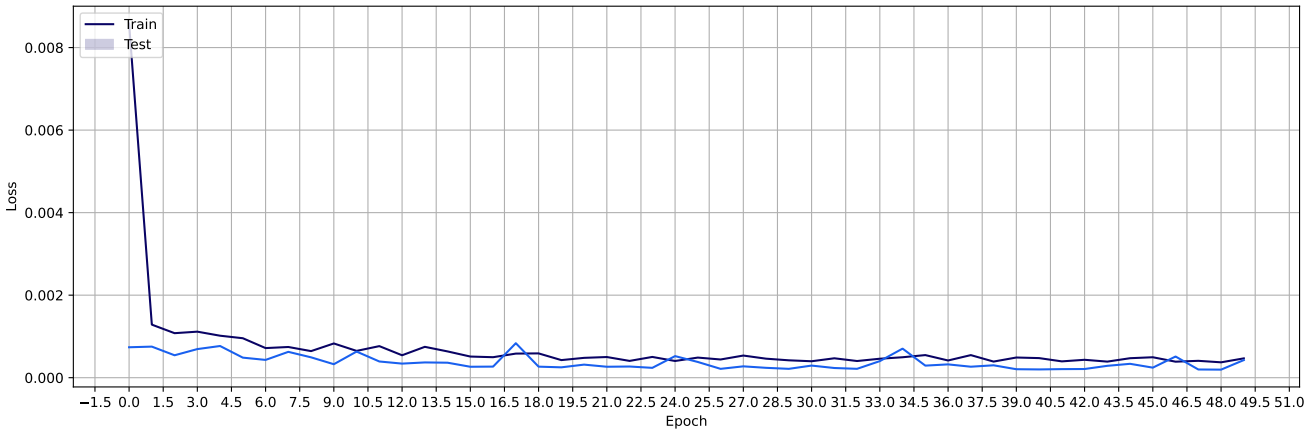


Figure 4. F-LSTM model loss.

6.2.2. Comparison of computation time between *F-LSTM* and LSTM

The amount of time needed to complete calculations and activities connected to developing, testing or generating forecasts using LSTM cryptocurrency price prediction models is referred to as computation time. It shows how long the underlying hardware and software took to do the necessary computations. Figure 5 shows the comparison of the computation time between the model developed for the whole data set and the client models used for federated learning, while, Figure 6 shows the prediction accuracy of the proposed framework. In Figure 5, the x-axis indicates the 2 different types of models and the y-axis indicates the computation time in seconds. In Figure 6, the x-axis indicates the timeline, and the y-axis indicates the actual price and prediction price for training and testing periods. Figure 5 clearly shows that the computation time utilized by the client models portrayed blue bar is significantly less than the computation time utilized by the model trained on the whole data set, represented by the green bar. It is also clear that the actual and predicted prices are almost similar, implying that the model has a high accuracy.

This is due to the distribution of data for the training of the local models. The time frame of the data used for the development of local models is 1 year for each client, whereas, the time frame of the data used to train the model trained on the whole data set is 5 years. More data would lead to a higher computation time, whereas fewer data would lead to a lower computation time. This makes FL a better and more efficient approach than the traditional one, as it saves computational time and resources. Hence, In federated learning, training is preferred as it is decentralized and carried out on client devices, allowing numerous devices to execute model updates in parallel. On the other hand, typical training involves training on the full dataset on a single machine, which can be slower due to the sequential processing of the data.

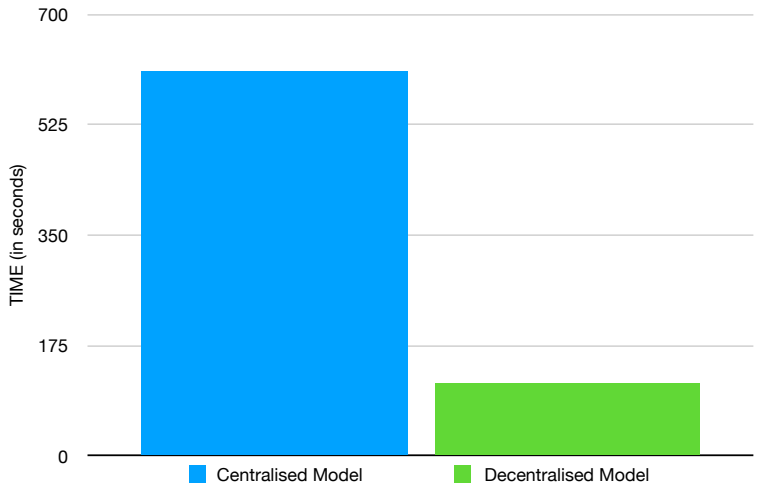


Figure 5. Comparison of computation time.

Figure 6. Prediction accuracy.

Another crucial element affecting the calculation time comparison between LSTM and F-LSTM, in addition to the data distribution and training durations, is the model update procedure. The communication overhead between the central server and clients is minimized with F-LSTM since model updates take place locally on client devices. This more efficient communication is particularly useful when working with clients with poor or limited connectivity. Traditional LSTM, on the other hand, uses centralized model updates and trains the entire model on a single machine. When working with huge datasets, this centralized processing can result in significant communication overhead. Additionally, the decentralized nature of F-LSTM enables asynchronous updates, allowing clients to update their models and speeding up training independently. These nuances in the update mechanism contribute to F-LSTM's superior computation time efficiency, making it a favorable choice for scenarios where efficient decentralized processing is imperative.

6.2.3. Performance under different optimizers

Optimizers are algorithms or methods used to modify the LSTM model's parameters (weights) while it is being trained. An optimizer's goal is to reduce the selected loss function by iteratively changing the model's parameters (weights) in accordance with the loss's gradients with respect to those weights. Figures 7 and 8 show the training and validation loss curve under different optimizers while training the proposed framework. In Figures 7 and 8, the x-axis represents the number of epochs and the y-axis denotes the measure of loss. The different colored curves are for the different optimizers. The different optimizers used are SGD, ADAGrad, RMSProp, Nadam and Adam. Adam gives the lowest loss and was used in the proposed framework.

On the basis of the gradients' average first and second moments, Adam changes the learning rate for each weight and bias parameter. Due to this, Adam is able to adjust the learning rate for each parameter, which results in faster convergence and better generalization. SGD is a basic optimizer that modifies the weights in accordance with the gradient of the loss function, but it has a slow convergence rate and is prone to local minima. Similar to Adam, Adagrad adjusts the learning rate for each parameter, but over time, it accumulates the square of the gradients, which may result in an excessively rapid decrease in the learning rate. The learning rate is also regulated by RMSProp, but it does so using a moving average of the squared gradients. Adam performs well for the proposed model for price prediction because it adapts the learning rate on a per-parameter basis, which leads to faster convergence and better generalization than other optimization algorithms.

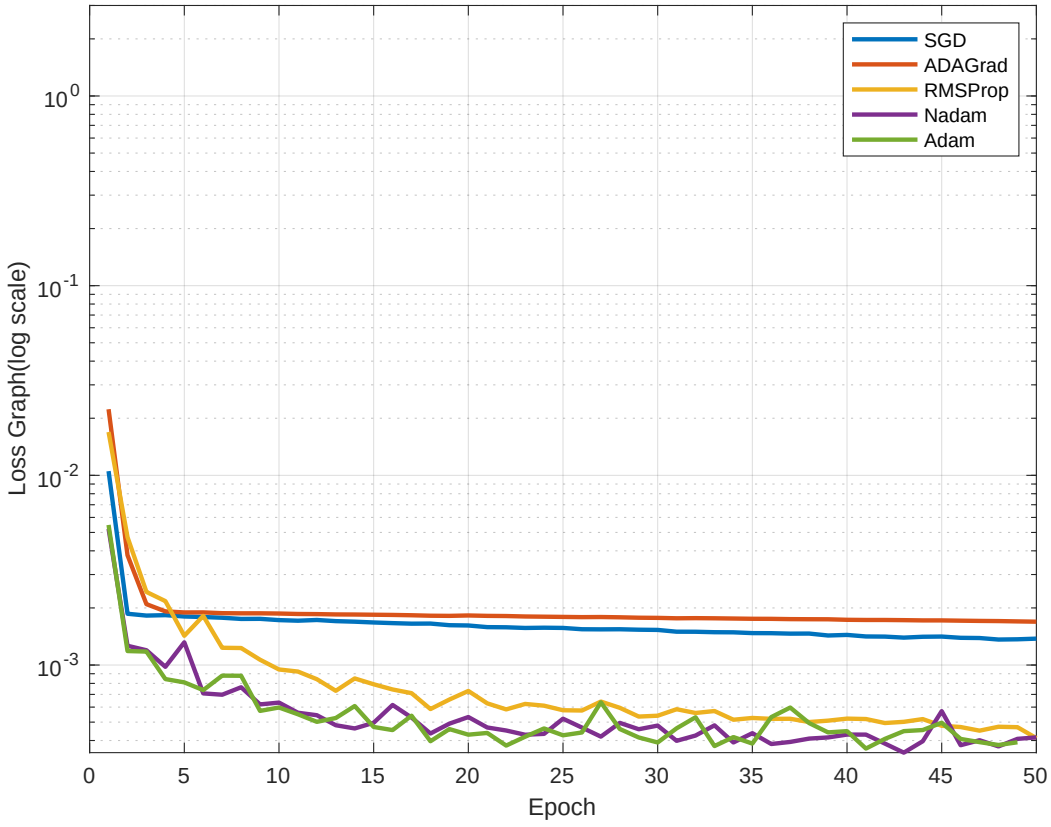


Figure 7. Loss for training under different optimizers.

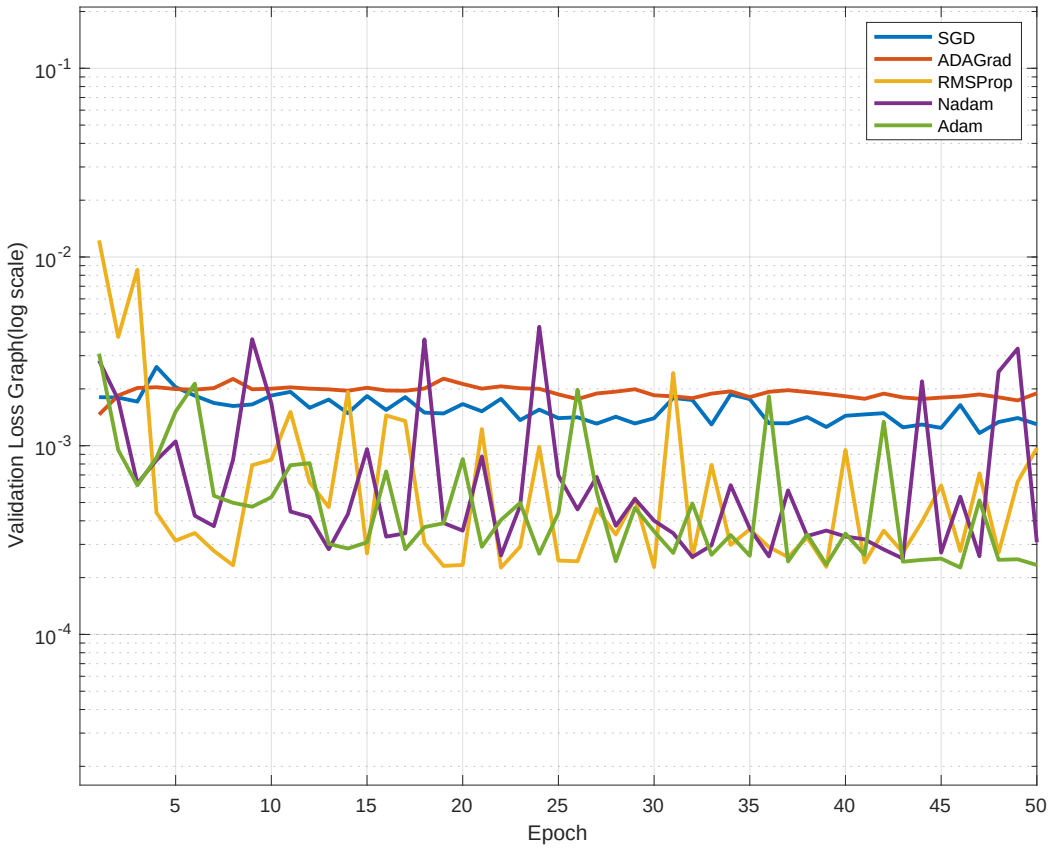
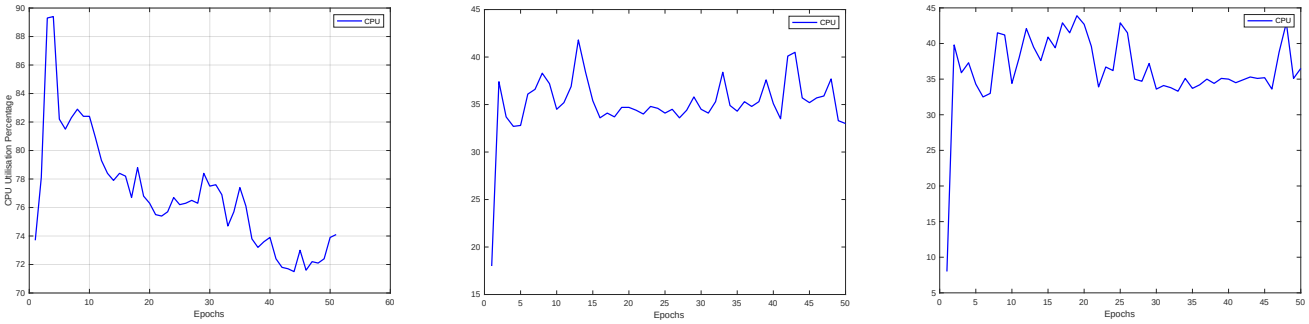


Figure 8. Validation loss for training under different optimizers.

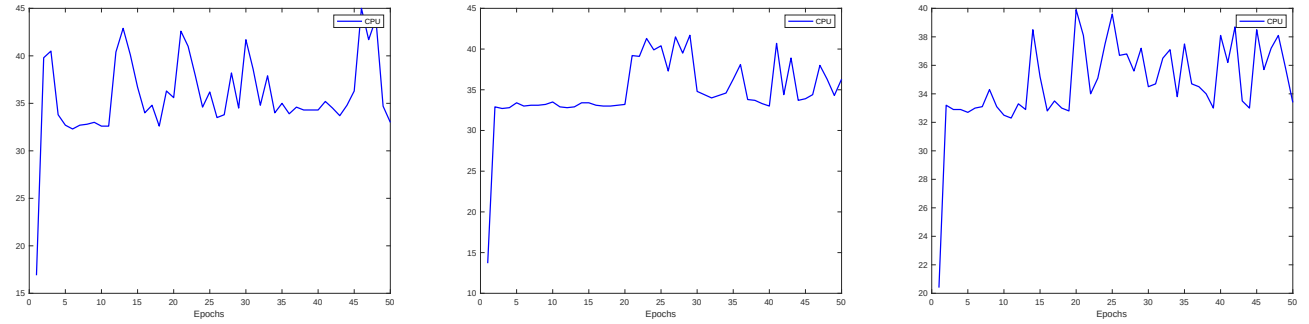
6.2.4. Comparison of CPU utilization between *F-LSTM* and LSTM

CPU utilization refers to the metric used to quantify how much the central processing unit (CPU) is used when the model-related computations are carried out. It calculates the proportion of time the CPU is used to do LSTM model-related operations, such as training, evaluating, or making predictions. Figure 9(a)–(f) show the comparison of the CPU utilization between the model developed for the whole data set and the local models used for federated learning. In the figures, the x-axis indicates the time intervals and the y-axis indicates the CPU utilization. It is evident that the local models trained for FL utilize significantly less computation power than the model trained on the whole data. The CPU utilization for training the client models is almost half of the CPU utilization during normal training.

In contrast to a normal centralized training situation, the client models in FL often execute a small amount of computation. This is a way that each client can undergo training using a smaller subset of the data rather than the complete dataset when employing federated learning. The quantity of data transport and processing required is lowered since the client models submit only their updated model parameters (weights) to the central server as opposed to providing the whole dataset. Additionally, before sending the model weights to the central server, FL involves compressing them. As a result, the amount of data provided is smaller, which also lowers the computational demands on the client models. Hence, FL training uses only a fraction of the computational resources used during normal training.



(a) Resource utilization for dataset. **(b)** Resource utilization for client 1. **(c)** Resource utilization for client 2.



(d) Resource utilization for client 3. **(e)** Resource utilization for client 4. **(f)** Resource utilization for client 5.

Figure 9. Performance of resource utilization for clients and normally trained model.

The resource utilization in traditional centralized training is significantly high, at about 74%. This is expected given that all computations, model updates and data processing are done on a single system, which takes a lot of resources. On the other hand, the FL method shows much greater resource efficiency. For the training of the 5 client models, roughly about 34% of the CPU resources were utilized. This significant decrease in resource use is a direct result of FL's decentralized structure. Through localized model updates on each client's own data, FL enables parallel processing and decreased resource requirements for each client. This fits with the fundamental ideas of distributed computing when analyzed theoretically. FL makes use of the parallelism that comes with training many models over distributed nodes, which leads to greater efficiency in resource management. This finding strongly suggests that the FL method is more resource-efficient than conventional centralized training. It not only lessens the pressure on individual devices but also highlights the possibility for large computational cost reductions, which makes FL an appealing option, particularly in situations where resource limitations are an issue. Thus, the results from Figure 9 offer strong support in proving that the FL scheme outperforms conventional training in terms of resource utilization.

6.2.5. Comparison of memory utilization between *F-LSTM* and LSTM

In the context of LSTM cryptocurrency price prediction models, the term memory utilization refers to the metric used to evaluate how much memory or memory resources were used during model training, evaluation or prediction. The memory needs of the LSTM model and related calculations are quantified. Figure 10 shows the comparison of the memory utilization between the model developed

for the whole data set and the local models used for federated learning. In the figures, the x-axis indicates the different models and the y-axis represents the memory utilization. It is evident that the local models trained for FL utilize significantly less memory than the model trained on the whole data, as their file sizes are significantly smaller.

As FL uses a distributed training approach that involves training multiple models on different subsets of the data rather than one model on the entire dataset, client models trained for FL significantly use less memory and have smaller file sizes than models trained on the entire dataset. A small portion of what would be needed to train a centralized model is used by each client device to train its model. In comparison to the centralized model that would be trained on the entire dataset, this leads to smaller models that utilize less memory and have smaller file sizes. Furthermore, FL models are often created to be compact and optimized for client devices with low resource requirements.

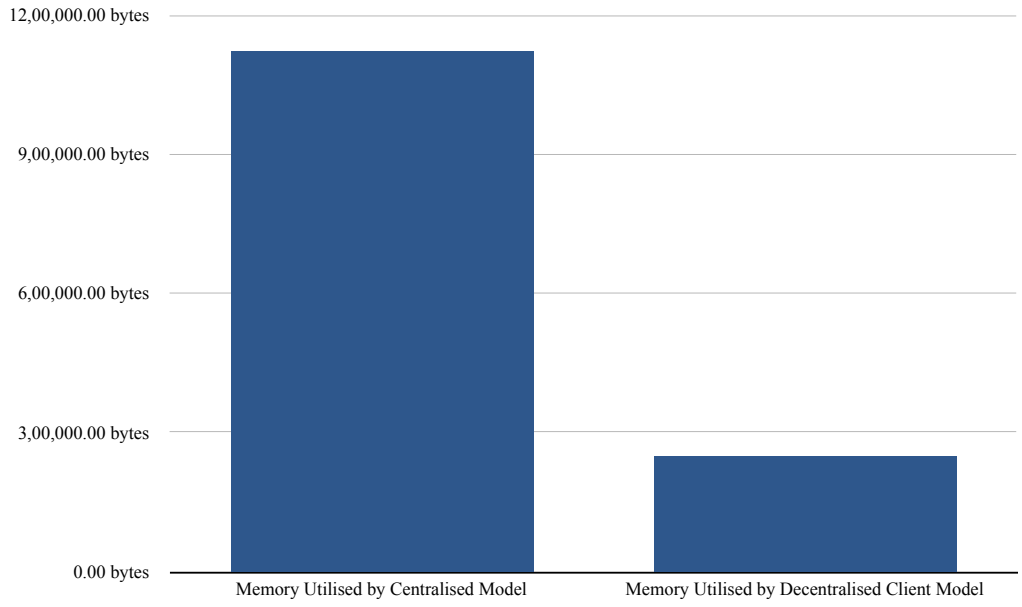


Figure 10. Memory utilization comparison.

6.2.6. Comparison with other FL schemes

Federated learning has emerged as a pivotal paradigm in the field of distributed learning for training models across decentralized clusters of nodes and updating model parameters in an optimized way. Within this realm of distributed learning, the three prominent averaging algorithms for updating global model weights are FedAvg, FedSGD and FedDyn, These schemes propose unique alignment for updating and averaging the client weights for global model learning. The respected schemes can be understood as follows.

1) *FedAvg*: The central server calculates the weighted average of the received updates from all the selected devices. Each update is weighted by the number of data samples that the device used for training. Devices with larger datasets contribute more significantly to the global model. The mean average percentage error while training the global model using FedAvg is 1.34% (as shown in Figure 11).

2) *FedSGD*: The averaging is done exactly the same as that of FedAvg but while updating the global model, that is done using the aggregated gradients. This update is performed using the stochastic gradient descent (SGD) algorithm or its variants, where a learning rate controls the step size of the update. The mean average percentage error while training the global model using FedSGD is 12.55%.

3) *FedDyn*: Instead of sending gradients or model updates directly, devices in FedDyn share information about the dynamics or changes in their local models. This information might include parameters like momentum, velocity or other relevant statistics. These Dynamics are used instead of the weights of the models for global model updation. The mean average percentage error while training the global model using FedDyn is 12.04%.

The wide difference in losses that occurred is particularly because of the notion in which the algorithm is applied. As for sequential learning parameters, the FedAvg performs better than others because of the weighted aggregation of client servers paired up with *Adam* optimizer, which catalyzes the process of finding the minima for the cost function while training the model.

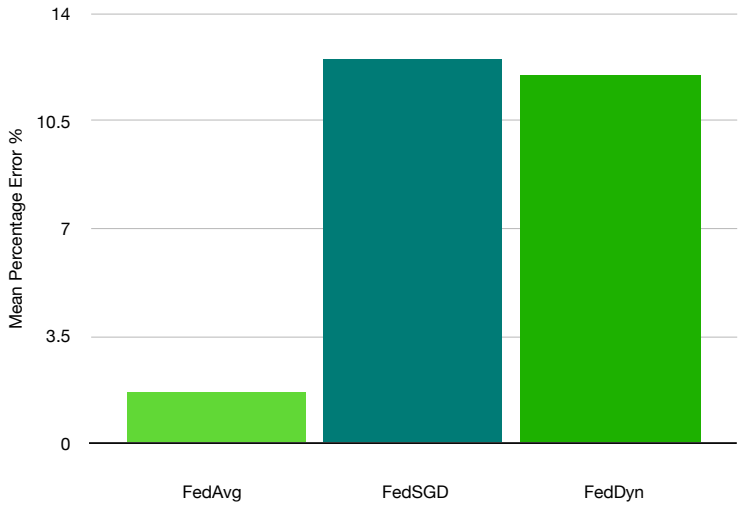


Figure 11. Comparison between different FL schemes.

6.2.7. Error measures

Assessing the effectiveness and reliability of any machine learning or statistical model requires measuring its prediction accuracy with high precision. Hence, researchers rely on error measures such as loss measures or performance metrics in their analysis. By quantifying the variance between actual values and predicted ones these metrics enable us to assess how accurate and reliable a given model's performance is getting over time. After training the global model for 10 epochs, the *F-LSTM* model was evaluated using three error measures which are, Median Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and Median Absolute Percentage Error (MDAPE). The model's Median Absolute Error (MAE) score is 91.98 percent, which means that on average, the predictions differ marginally from the measured data. This shows that the model performs fairly well in terms of absolute error.

The model's ability for precise prediction improves with decreasing MAE. The Mean Absolute Percentage Error (MAPE) of 6.23 percent shows that the model's predictions often deviate from the actual values by about that percent. This portrays that, given the size of the target variable, the model's performance is fairly accurate. A lower MAPE indicates that the model is more capable of making accurate predictions. The Median Absolute Percentage Error (MDAPE) of the model is 5.1 percent. This demonstrates the model's accuracy since a lower MDAPE implies greater prediction precision. The model's consistency in producing predictions that are close to the actual values illustrates both its dependability and its effectiveness. The model's effectiveness is demonstrated by the excellent results it produces after training on only 10 epochs. The model exhibits a streamlined and resource-efficient learning procedure by successfully resolving dependencies through small-batch training on the complete dataset for a limited number of epochs.

7. Conclusions and future works

In conclusion, this study marks a substantial advancement in the field of predicting cryptocurrency prices. Our innovative method completely transforms the predictive analytics environment in this dynamic and privacy-sensitive field by integrating LSTM with FL. Our approach takes advantage of the distributed data-gathering capabilities of FL to improve the pattern recognition performance of LSTM by collecting data from decentralized nodes while adhering to tight privacy measures. Without the requirement for centralization, this combination produces a richer dataset, which leads to remarkable efficiency throughout the training stage. Our methodology proves its ability to deliver extremely accurate cryptocurrency price projections, with training loss converging to an amazing minimum of 2×10^{-4} and validation loss resting at an amazing 2.3×10^{-4}.

Our federated integrated LSTM model also complies with current edge computing trends, making it more than just an advance in cryptocurrency price prediction. It improves operational efficiency and, more importantly, satisfies the fundamental principles of data privacy and security by executing computations closer to the sources of the data. As we shift our attention to a comparison with traditional approaches, like LSTM, ARIMA and GARCH models (all evaluated on the same dataset), the findings highlight the significant advantages of our strategy. Our approach not only performs admirably in terms of loss reduction, but it also demonstrates fantastic computing efficiency. It is crucial to understand the broader implications of our findings by looking beyond the metrics. Our approach gives traders, investors and financial analysts a strong tool to help them make judgments at a time when cryptocurrencies are having an increasing impact on financial markets. Furthermore, our focus on preserving data security and privacy in the age of data-driven finance is further demonstrated by our commitment to the federated method. Our study paves the way for more accurate, effective and privacy-respecting cryptocurrency price projections as we look into the future. It claims to revolutionize financial decision-making by providing a way to more trustworthy insights and well-informed actions in the intricate and constantly changing world of the cryptocurrency markets.

We are resolutely devoted to improving our *F-LSTM* framework on numerous fronts in future works, hoping to deliver solid answers and insights in the field of cryptocurrency price prediction in order to address the research deficiencies found in our current study. First, in order to give the model the dynamic versatility needed to thrive in the constantly shifting real-time cryptocurrency environment, we intend to optimize its integration with reinforcement learning. The accuracy and

responsiveness of cryptocurrency price projections will be improved by this improvement, which will enable our framework to take proactive actions in reaction to market swings. Second, we want to extend the F-LSTM schema's range of applications by using it with different cryptocurrencies, offering a sectorial and application-centric viewpoint. We seek to increase our understanding of how various cryptocurrencies behave and react to comparable prediction approaches by analyzing a variety of digital assets, each with its own distinct market dynamics. This will eventually lead to more specialized predictive models. Finally, we are looking into how split learning and lightweight encryption techniques can be applied to the analysis of cryptocurrency price data. Split learning has the potential to improve data security and privacy, particularly in a field where data protection is crucial. Also, applying lightweight encryption to the trained weight can enhance the security and privacy measures of the proposed work. This study demonstrates our dedication to increasing cryptocurrency markets' accuracy and data security by taking a diverse approach to comprehend better and forecast the prices of digital assets.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgements

This work was funded by the Researchers Supporting Project Number (RSP2023R509), King Saud University, Riyadh, Saudi Arabia.

Conflict of interest

The authors declare no conflict of interest.

References

1. K. Pilbeam, *Finance and Financial Markets*, Bloomsbury Publishing, 2018.
2. M. Watorek, S. Drozdz, J. Kwapien, L. Minati, P. Oswiecimka, M. Stanuszek, Multiscale characteristics of the emerging global cryptocurrency market, *Phys. Rep.*, **901** (2021), 1–82. https://doi.org/10.1016/j.physrep.2020.10.005
3. M. Resta, P. Pagnottoni, M. E. D. Giuli, Technical analysis on the bitcoin market: Trading opportunities or investors' pitfall, *Risks*, **8** (2020), 44. https://doi.org/10.3390/risks8020044
4. N. Miller, Y. Yang, B. Sun, G. Zhang, Identification of technical analysis patterns with smoothing splines for bitcoin prices, *J. Appl. Stat.*, **46** (2019), 2289–2297. https://doi.org/10.1080/02664763.2019.1580251
5. R. Parekh, N. P. Patel, N. Thakkar, R. Gupta, S. Tanwar, G. Sharma, et al., DL-Guess: Deep learning and sentiment analysis-based cryptocurrency price prediction, *IEEE Access*, **10** (2022), 35398–35409. https://doi.org/10.1109/ACCESS.2022.3163305

6. B. M. Henrique, V. A. Sobreiro, H. Kimura, Literature review: Machine learning techniques applied to financial market prediction, *Expert Syst. Appl.*, **124** (2019), 226–251. https://doi.org/10.1016/j.eswa.2019.01.012

7. L. Alessandretti, A. ElBahrawy, L. M. Aiello, A. Baronchelli, Anticipating cryptocurrency prices using machine learning, *Complexity*, **2018** (2018), 8983590. https://doi.org/10.1155/2018/8983590

8. T. E. Koker, D. Koutmos, Cryptocurrency trading using machine learning, *J. Risk Financ. Manage.*, **13** (2020), 178. https://doi.org/10.3390/jrfm13080178

9. N. P. Patel, R. Parekh, N. Thakkar, R. Gupta, S. Tanwar, G. Sharma, et al., Fusion in cryptocurrency price prediction: A decade survey on recent advancements, architecture, and potential future directions, *IEEE Access*, **10** (2022), 34511–34538. https://doi.org/10.1109/ACCESS.2022.3163023

10. D. H. Kwon, J. B. Kim, J. S. Heo, C. M. Kim, Y. H. Han, Time series classification of cryptocurrency price trend based on a recurrent LSTM neural network, *J. Inf. Process. Syst.*, **15** (2019), 694–706. https://doi.org/10.3745/JIPS.03.0120

11. E. S. Pour, H. Jafari, A. Lashgari, E. Rabiee, A. Ahmadisharaf, Cryptocurrency price prediction with neural networks of LSTM and bayesian optimization, *Eur. J. Bus. Manage. Res.*, **7** (2022), 20–27. https://doi.org/10.24018/ejbmr.2022.7.2.1307

12. S. P. Ardakani, N. Du, C. Lin, J. C. Yang, Z. Bi, L. Chen, A federated learning-enabled predictive analysis to forecast stock market trends, *J. Ambient Intell. Hum. Comput.*, **14** (2023), 1–7. https://doi.org/10.1007/s12652-023-04570-4

13. W. Jiang, Applications of deep learning in stock market prediction: Recent progress, *Expert Syst. Appl.*, **184** (2021), 115537. https://doi.org/10.1016/j.eswa.2021.115537

14. W. An, L. Wang, D. Zhang, Comprehensive commodity price forecasting framework using text mining methods, *J. Forecasting*, **42** (2023), 1865–1888. https://doi.org/10.1002/for.2985

15. Q. Zhang, C. Qin, Y. Zhang, F. Bao, C. Zhang, P. Liu, Transformer-based attention network for stock movement prediction, *Expert Syst. Appl.*, **202** (2022), 117239. https://doi.org/10.1016/j.eswa.2022.117239

16. S. Halder, FinBERT-LSTM: Deep learning based stock price prediction using news sentiment analysis, *arXiv preprint*, (2022), arXiv:2211.07392. https://doi.org/10.48550/arXiv.2211.07392

17. Z. Shi, Y. Hu, G. Mo, J. Wu, Attention-based CNN-LSTM and XGBoost hybrid model for stock prediction, *arXiv preprint*, (2023), arXiv:2204.02623. https://doi.org/10.48550/arXiv.2204.02623

18. P. Nithyakani, R. J. Tom, P. Gupta, A. Shanthini, V. M. John, V. Sharma, Prediction of bitcoin price using Bi-LSTM network, in *2021 International Conference on Computer Communication and Informatics (ICCCI)*, Coimbatore, India, (2021), 1–5. https://doi.org/10.1109/ICCCI50826.2021.9402427

19. P. Bhattacharya, S. B. Patel, R. Gupta, S. Tanwar, J. J. P. C. Rodrigues, SATYA: Trusted bi-lstm-based fake news classification scheme for smart community, *IEEE Trans. Comput. Social Syst.*, **9** (2022), 1758–1767. https://doi.org/10.1109/TCSS.2021.3131945

20. C. Luo, L. Pan, B. Chen, H. Xu, Bitcoin price forecasting: An integrated approach using hybrid LSTM-ELM models, *Math. Probl. Eng.*, **2022** (2022), 2126518. https://doi.org/10.1155/2022/2126518

21. I. E. Livieris, N. Kiriakidou, S. Stavroyiannis, P. Pintelas, An advanced CNN-LSTM model for cryptocurrency forecasting, *Electronics*, **10** (2021), 287. https://doi.org/10.3390/electronics10030287

22. F. Ferdiansyah, S. H. Othman, R. Zahilah Raja Md Radzi, D. Stiawan, Y. Sazaki, U. Ependi, A LSTM-method for bitcoin price prediction: A case study yahoo finance stock market, in *2019 International Conference on Electrical Engineering and Computer Science (ICECOS)*, (2019), 206–210. https://doi.org/10.1109/ICECOS47637.2019.8984499

23. Pip install yfinance, YFinance 0.2.21, pip install yfinance, Available from: https://pypi.org/project/yfinance/.

24. L. Peng, L. Wang, D. Xia, Q. Gao, Effective energy consumption forecasting using empirical wavelet transform and long short-term memory, *Energy*, **238** (2022), 121756. https://doi.org/10.1016/j.energy.2021.121756

AIMS Press