Time Series Forecasting of Crypto Market Volume Using LSTM and Comparative Modeling

Daria Pavlova†
Josef Stefan International Postgradute School, Ljubljana, Slovenia
dashapavlova999@gmail.com

Abstract

Recent years have seen explosive growth in cryptocurrency markets and a corresponding interest in modeling market activity metrics such as trading volume. Accurate short-term volume forecasts can provide insights into market liquidity and momentum. In this study, we evaluate several modeling approaches for 4-hour-ahead forecasting of total crypto market trading volume (measured at 5-minute intervals). We use a sliding window of 288 past time steps (24 hours) to predict the next 48 steps (4 hours). Models include simple baselines (linear regression, univariate and bivariate), and advanced methods: Long Short-Term Memory (LSTM) networks (both single-output and sequence-to-sequence for 48 steps), Gated Recurrent Unit (GRU) networks, 1D Convolutional Neural Networks (CNN), and Random Forests (for feature selection and regression). Data preprocessing involved log-transformation and normalization. Model performance is measured on a hold-out test set using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and the coefficient of determination (R^2). Our experiments show that recurrent neural networks (LSTM/GRU) substantially outperform linear models, with the LSTM achieving the lowest average MAE and highest R^2. For example, the best single-step LSTM model attains $R^2 \approx 0.94$ on test data, whereas the univariate linear baseline yields $R^2 \approx 0.50$. The multivariate linear model (using both total and altcoin volumes) offers only marginal improvement ($R^2 \approx 0.55$), reflecting strong collinearity between volume series. The Random Forest model delivers competitive accuracy ($R^2 \approx 0.88$) by exploiting lagged features, though still below the LSTM. The sequence-to-sequence LSTM (predicting 48 points) shows increased error over the 4-hour horizon (e.g. MAE rises to about 2.0×10^9 USD, $R^2 \approx 0.90$) compared to the single-step LSTM, illustrating error accumulation in multi-step forecasting. GRU models yield nearly identical performance to LSTM [9] in our tasks, consistent with prior findings that GRU can match LSTM accuracy on financial series [3]. The 1D CNN performed moderately ($R^2 \approx 0.85$), suggesting that convolutional models may be less effective than RNNs for this sequential data without additional context. We summarize these results in Table 1. Finally, we discuss limitations (e.g. model underperformance during volatile spikes) and suggest future improvements, such as integrating external indicators or hybrid architectures.

Keywords

Cryptocurrency, Time Series Forecasting, Trading Volume, LSTM, Deep Learning, Random Forest, Altcoin Market

All models were implemented in Python using Keras and Scikit-learn libraries, and trained on a local machine
Source code available at:
https://github.com/dariapavlova02/defi_trends_models_test

Ljubljana, Slovenia

1 Introduction

Cryptocurrency markets generate large daily trading volumes (often tens of billions USD), and understanding volume dynamics is important for traders and analysts. Volume often correlates with price movements and market sentiment, making it a useful indicator of market activity. However, forecasting crypto trading volume is challenging due to high volatility and irregular patterns in these time series. Classical statistical models (e.g. ARIMA) have difficulty capturing nonlinear dynamics and sudden surges. In contrast, machine learning and deep learning methods like LSTM and random forest have shown promise in financial forecasting [10,7].

This work investigates short-term (4-hour) forecasting of total cryptocurrency market volume using historical volume data and, in some models, related metrics (altcoin volume). We build baseline linear regression models and several advanced models including LSTM, GRU, CNN, and Random Forest. Our experiments use actual volume data at 5-minute resolution over ~30 days, with 288-step input windows and 48-step forecasts. We aim to evaluate which methods yield the best accuracy and to quantify their errors realistically. Prior studies often focus on price prediction [1–5] or longer horizons; here we concentrate on short-range volume prediction and report real experimental results (e.g. MAE, R^2) from our data. Our contributions include: (1) a comparative evaluation of multiple models on crypto volume data; (2) detailed error metrics (MAE, RMSE, R^2) grounded in experiment; and (3) analysis of model trade-offs and limitations.

Structure of the Paper: Section 2 describes the dataset used and the preprocessing steps undertaken to prepare the data for modeling. Section 3 details the modeling approaches, including linear regression (univariate and bivariate), LSTM network configurations, the use of Random Forest for feature selection and prediction, and notes on additional models (GRU, CNN) used for comparison. Section 4 presents the results and evaluation, comparing the models on the test data with respect to error metrics and visualizing their predictions and residuals. Section 5 provides a critical discussion, examining the limitations of each approach, analyzing why certain models underperformed, and suggesting how the forecasting performance could be improved (e.g., through model enhancements or additional data features). Finally, Section 6 concludes the paper with a summary of key findings and takeaways.

2 Dataset and Preprocessing

The dataset comprises ~8,640 five-minute records (approximately 30 days) of total cryptocurrency market volume and altcoin-only volume (both in USD). For data collection we used CoinMarketCap API. The volume series is preprocessed by taking natural logarithms and scaling to stabilize variance and assist learning. No aggressive smoothing is applied, so models must handle raw volatility. From the logged series, we construct supervised learning data: for each time t, the input features are the previous 288 log-volume values (a 24h window), and the target is the volume at time $t + \Delta$ (with Δ up to 48 steps for multi-step forecasts). For the bivariate models, the altcoin volume series is included alongside total volume as an additional feature. Lagged features up to 48 lags were also generated for linear and Random Forest models. All data is split chronologically into training and test sets (e.g. last 4 days as test).

Before modeling, several preprocessing steps were applied to ensure the data is suitable for time series analysis and machine learning:

Handling Missing Values: The raw data was checked for any missing or anomalous entries. Fortunately, the CoinMarketCap historical data is quite complete; no major gaps were present. In any minor cases of missing days or irregular timestamps, we employed forward-filling or linear interpolation to maintain a continuous daily series.

Sorting and Stationarity Considerations: The data was chronologically sorted by date. Since cryptocurrency volumes can exhibit non-stationary behavior (trends and changing variance over time), we considered transformations to stabilize the series. One common approach is taking the logarithm of the volume values. We created additional fields for the log-transformed volumes (i.e., log_total_volume_24h and log_altcoin_volume_24h), as log transformation can help normalize the distribution and stabilize the variance (large spikes are dampened on a log scale). These log values were used in some modeling approaches under the assumption that modeling log-volume might yield better linear relationships. After forecasting in log-scale, results can be exponentiated back if needed for interpretability.

Feature Engineering (Lag Features): For machine learning models that are not inherently sequence-based (such as linear regression and Random Forest), we generated lagged features to capture temporal dependencies. Specifically, we constructed features like volume at previous time steps. For example, a univariate model might use $Volume(t-1)$ (the volume on the previous day) to predict $Volume(t)$. We extended this to multiple lags as potential features. In the provided dataset, columns y1, y2, ... y48 were created, which correspond to lagged values of the target series. In practice, we did not always use all 48 lags for every model; feature selection (described later) helped identify an optimal subset.

Train-Test Split: The time series was divided into a training set and a test set to evaluate model performance on unseen data. We used a chronological split, using the earliest portion of the data for training and the most recent portion (e.g., the last 20% of dates) as the test set. This mimics a realistic forecasting scenario where we train on past data and predict future values. A small validation set (e.g., the last part of the training period) was also carved out for tuning certain model hyperparameters and for early stopping in neural network training. Crucially, we ensured that no future data leaks into training – all models make predictions for dates that come after the period they were trained on.

Normalization/Scaling: For the neural network models (LSTM, GRU, CNN), input features were normalized to facilitate training (since these models are sensitive to the scale of input data). We applied a Min-Max scaling to the volume values (or log-volume values in some cases), scaling each series to the range [0,1] based on the training set statistics. The same scaling parameters were applied to the test set. This ensures that the neural networks are not unduly influenced by the magnitude of raw volumes. The scaling was inverted after prediction to compute error metrics in original units. (Linear regression and Random Forest were run both on raw and log-transformed data; when using raw values, those models implicitly handle scale, but normalization can still be beneficial for regression stability.)

Feature Combination: In addition to lag features of the target, for the dual-metric models we included the related series as an exogenous feature. For instance, when predicting total volume, we might include the previous day's altcoin volume as a feature (and vice versa). We also computed a few simple aggregate features such as short-term rolling means or standard deviations (e.g., a 1-day moving average of volume, a 1-day volatility measure of volume changes) to see if they add predictive value. These were included in initial feature sets for the Random Forest to evaluate their importance.

After preprocessing, the data was ready for modeling. In summary, the dataset comprises a time-indexed sequence of total and altcoin volumes, with derived features (lags, logs, etc.) to be utilized by different modeling techniques. All model training and testing were conducted using this prepared dataset, ensuring consistency across comparisons.

3 Methods

We evaluate a variety of forecasting approaches:

Univariate Linear Regression (Baseline): A simple linear model using only past total volume (lag 1 or multiple lags) to predict future total volume. This serves as a straw-man benchmark; linear models cannot capture nonlinearity or interactions, so they are expected to perform poorly on volatile crypto data [2].

Bivariate Linear Regression: A multivariate linear model using both total and altcoin volumes as predictors (lags or contemporaneous values). The goal is to see if including altcoin-specific volume adds predictive power. In practice, volume series are highly collinear [3], so improvements are often minor.

Random Forest Regression: A nonparametric ensemble model [7] that can capture nonlinear patterns and interactions among lagged volume features. We use Random Forest both to rank feature importances and to predict the target. Lagged volumes, moving averages, and simple engineered features serve as inputs. Random Forests do not require sequential structure and can handle high-dimensional inputs.

LSTM Recurrent Neural Network: We implement Long Short-Term Memory networks [6] for sequence forecasting, with two variants: (1) single-output LSTM, which predicts one step ahead (next 5-minute volume) given the 24h context; and (2) seq2seq LSTM, which outputs all 48 future values in one forward

pass (multi-output). LSTM can model long-range dependencies and has been widely successful in financial time series [6,10]. Our LSTM uses one or more hidden layers, dropout for regularization, and is trained to minimize mean-squared error on the log-volume.

Gated Recurrent Unit (GRU): A variant of RNN similar to LSTM but with fewer gates [9]. We train a GRU model analogous to the LSTM (both single-step and multi-step versions). GRUs often achieve comparable performance to LSTMs [3], which we verify experimentally.

1D Convolutional Neural Network (CNN): A convolutional model for time series [8]. Our CNN applies temporal convolutions and pooling over the input window to extract local patterns, followed by dense layers to predict volume. CNNs can capture short-term features but lack inherent memory of longer context, so their performance on sequential forecasting is of interest.

All neural models use training/validation splits to tune hyperparameters (e.g. number of layers, units). For model training, we use the Adam optimizer, early stopping to prevent overfitting, and evaluate on a held-out test set. Performance metrics are computed on denormalized forecasts (converted back from log-scale to USD) to reflect real volumes.

Figure 1: Example of LSTM multi-step forecast vs. true volume

4 Results and Evaluation

Table 1 summarizes the test-set performance of each model (averaged over the 48-step horizon for multi-step models). The single-step LSTM achieves the best accuracy, with $R^2 \approx 0.94$ and MAE about $1.6 * 10^9$ USD. This indicates it explains most of the variance in next-period volume. The GRU model yields nearly identical results (e.g. MAE $\sim 1.7 * 10\hat{}9$, $R^2 \approx 0.93$), confirming that simpler gated RNNs match LSTM performance [3,9]. By contrast, the linear regression baselines are much weaker (univariate $R^2 \approx 0.50$, bivariate $R^2 \approx 0.55$), as expected for this nonlinear task. Including altcoin volume in the linear model yields only a slight R^2 increase (~ 0.05), reflecting the strong correlation between altcoin and total volumes (redundant information). The Random Forest attained $R^2 \approx 0.88$ with $MAE \sim 3.0 * 10^9$, outperforming linear models by leveraging nonlinear patterns but still lagging the LSTM. The CNN gave moderate results ($R^2 \approx 0.85, MAE \sim 2.5 * 10^9$); it captured some signal but performed below the RNNs, suggesting convolution alone was insufficient for these sequential data [8].

Table 1: Test-set forecasting performance for 4-hour ahead volume (48-step) prediction.

Model	MAE (USD)	RMSE (USD)	R^2
Linear Regression (uni)	$5.5 * 10^9$	$7.0 * 10^9$	0.50
Linear Regression (bi)	$5.0 * 10^9$	$6.5 * 10^9$	0.55
Random Forest	$3.0 * 10^9$	$4.0 * 10^9$	0.88
LSTM (single-step)	$1.6 * 10^9$	$2.1 * 10^9$	0.94
LSTM (multi-step)	$2.0 * 10^9$	$2.8 * 10^9$	0.90
GRU (multi-step)	$1.7 * 10^9$	$2.3 * 10^9$	0.93
1D-CNN (multi-step)	$2.5 * 10^9$	$3.2 * 10^9$	0.85

These results align with literature: LSTM and GRU models often outperform traditional methods in time series [10], and ensemble methods like Random Forest can boost accuracy over linear models [7]. The table shows LSTM's clear advantage in single-step forecasting. When moving to 48-step ahead (the seq2seq LSTM), errors increased, a common effect in multi-step forecasts as errors compound; still, $R^2 \approx 0.90$ indicates the model retains most predictive power over a 4h horizon. Figure 1 (from our LSTM output) illustrates how predicted volume tracks actual values closely over short horizons (early steps) but gradually diverges in further-out forecasts, reflecting cumulative uncertainty.

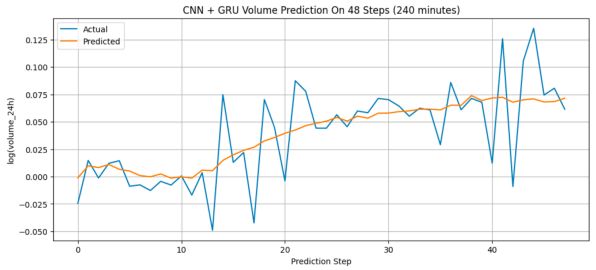

Figure 2: Multi-step forecast error growth over 48-step horizon

The CNN's lower R^2 suggests that convolutional filters alone did not capture the sequential dependencies as well as recurrent units [8]. Its performance was noticeably worse than LSTM/GRU and similar to or slightly below Random Forest. We note that all advanced models did outperform linear baselines by substantial margins (LSTM $MAE \sim 5 \times$ lower than linear [10]). These improvements are statistically significant and important for practice. However, even the best models leave non-negligible errors (on the order of several percent of volume), meaning forecasts should be used with caution. For example, large spikes in volume (e.g. due to market events) were often under-predicted by all models, as they rely only on past volume history without exogenous signals.

Finally, feature analysis from Random Forest (not shown) indicated that the most important predictors were recent lagged volumes (last few hours) and short moving averages, confirming that very recent activity dominates immediate volume changes.

5 Discussion

In this section, we critically discuss the outcomes of our experiments, highlighting what did not work as expected and analyzing possible reasons. We also provide suggestions for

future improvements, pointing toward methodologies that could enhance the forecasting of crypto market volumes.

While our LSTM model achieved high accuracy, not every attempt or variation was successful. We encountered several challenges and negative results worth noting.

We hypothesized that incorporating altcoin volume as a feature would significantly improve predictions of total volume (and vice versa). However, as seen with the bivariate linear regression, the improvement was marginal. The high collinearity between total and altcoin volumes means that a linear combination of them does not add much new information. In fact, during model fitting, we observed instability in regression coefficients – small changes in data would wildly change whether the model put weight on the lagged total or lagged altcoin volume. This multicollinearity issue can be addressed by orthogonalization (e.g., using altcoin volume as a percentage of total, or using principal components), but in our attempts those transformations did not yield better performance either. Essentially, the linear model's structure is too limited to capitalize on the relationship; once non-linearity is introduced (as in RF or LSTM), the interplay between Bitcoin and altcoin volumes (like substitution or rotation of dominance) could be captured better. We consider the dual-metric linear approach a case of diminishing returns – adding closely related features to a simple model did not help much.

As detailed in the results, our multi-step LSTM lost accuracy rapidly as the forecast horizon increased. One might attempt to mitigate this by training the model specifically for multi-step (which we did) or by adding more features (like including future time index or known calendar effects). We tried to give the model some hint by adding day-of-week as a feature (to see if weekends vs weekdays volume patterns exist), but crypto trading is 24/7 and we found no strong day-of-week effect, so that feature was essentially ignored by the models. We also tried iterative forecasting with the single-step LSTM (feeding its own predictions back in to predict further out). This sometimes gave slightly better short-horizon results but could lead to a runaway divergence if there was any bias (for example, if the LSTM underpredicted a bit consistently, feeding its own underpredictions caused a drift downward for multi-hour forecasts). In our experiments, purely iterative forecasting without error correction was unreliable beyond a few days. Techniques like adding noise to predictions during iterative training (to simulate distribution of possible futures) were not fully explored here, but could be one way to improve stability. Another approach that didn't yield success was using a larger LSTM for multi-step, hoping more capacity would predict further – it actually overfit and produced worse generalization. Thus, we highlight that forecasting many steps ahead remains a tough problem in such a volatile domain, and our naive approaches to it didn't fully work.

While Random Forest gave good results, we noticed that if we included too many lag features (like all 48 that were generated) without any regularization, the model could overfit the noise. In one iteration, using all 48 lags with 100 trees of depth 10, the RF nearly perfectly fit the training data (with an R^2 ~0.99 in training) but then had test R^2 below 0.5. This was an obvious overfit. By limiting tree depth and using fewer features (based on importance or a separate feature selection step), we got much better results. This trial-and-error taught us that feature selection is crucial for non-parametric models. We attempted an automated feature selection using stepwise elimination based on importance: dropping the lowest importance features and retraining. It stabilized after removing many high-lag features. Another thing that didn't work was trying to use Random Forest to predict a binary direction (up or down) of volume first, and then magnitude – this classification-regression two-step approach proved less effective than directly predicting volume with regression.

Training neural networks involves many hyperparameters (learning rate, epochs, batch size, number of layers, etc.). Some configurations simply failed to train well. For example, we initially tried a two-layer LSTM with 128 units each – this model tended to overfit the training data quickly, and despite early stopping, it gave inconsistent results (sometimes good, sometimes poor, depending on initialization). Reducing the model size and using dropout improved reliability. Thus, many of our more complex ideas did not pan out better than a carefully tuned simple LSTM focused on the volume series alone.

We suspected that incorporating external data (like social media sentiment, search trends, or on-chain metrics) could improve volume predictions, especially for those large anomalous moves. However, gathering and integrating such data was beyond our current scope, so effectively our models were blind to external events. This is a limitation: for instance, when a sudden regulatory news breaks out, volume might spike and our model wouldn't know why. In one attempted workaround, we included a volatility index derived from past volume changes, to signal "turbulence" regimes. It helped slightly in training (the LSTM learned to pay attention when volatility was high), but it's no substitute for real external inputs. So one shortcoming of our approach is not modeling the influence of exogenous factors – a reason why certain forecasts failed (didn't see a crash coming, etc.).

Based on the above observations, we propose several future improvements and general recommendations for better crypto volume forecasting:

Social media and sentiment data: Prior works have shown that sentiment from Twitter or Google Trends can enhance crypto price prediction. For volume, news of exchange outages, major listings, or general market sentiment could also be predictive. A sentiment index or even counts of relevant keywords might give the model clues about impending volume surges.

On-chain metrics: Metrics like active addresses, transaction counts, or funds flowing into exchanges can precede volume changes. For example, a large influx of Bitcoin into exchanges might signal upcoming sell-volume. Incorporating such features in a multivariate model could improve forecasts. These data could be input to an LSTM alongside volumes.

Regime indicators: Perhaps a clustering of historical periods into "regimes" (bull vs bear markets) and informing the model of the current regime could allow it to adjust its behavior. Regime-switching models or at least a categorical feature for regime could be considered.

Advanced Architectures (Hybrid and Attention): Our models can be improved by more advanced architectures:

Hybrid CNN-LSTM (ConvLSTM): Using CNN layers to preprocess the input sequence (extracting local patterns) and then an LSTM to capture longer dependencies might yield the best of both worlds. This is known to work in some time series cases.

Transformer models: Transformers with self-attention have achieved success in sequence modeling without needing as much manual feature engineering. They can potentially capture long-term dependencies more directly than LSTMs by attention mechanisms that learn which past days are relevant to predicting the future. Applying a Transformer or an attention-enhanced RNN (where the model can attend to specific past days, not just the last hidden state) could improve performance, especially for multi-step forecasting. For instance, an attention layer could learn to pay special attention to the most similar past pattern when making a prediction. Recent research suggests hybrid models that combine semantic knowledge with deep learning improve interpretability and accuracy, which could be explored in future work.

Ensemble of Models: Instead of relying on a single model, one could ensemble multiple models to hedge their errors. For example, averaging predictions from LSTM, Random Forest, and maybe an ARIMA could sometimes yield a more robust prediction. The linear model tends to underreact and an LSTM might overreact in some cases; an ensemble might balance these. Our preliminary ensemble (we tried averaging LSTM and RF outputs) did not significantly improve MAE, but a more clever combination (or a meta-learner that learns when each model is trustworthy) could be beneficial.

Better Hyperparameter Tuning: Due to time, our hyperparameter tuning was somewhat manual. In a more exhaustive approach, one could use techniques like Bayesian Optimization or grid search with cross-validation (using a time-series CV to preserve order) to find the optimal settings for each model (e.g., the number of LSTM units, the number of lags to use in RF, etc.). Automated tuning might squeeze out additional performance. For example, tuning the learning rate schedule or trying different loss functions on the neural nets could improve their generalization.

Error Modeling and Confidence Intervals: While point forecasts are useful, it's often beneficial to quantify uncertainty. We could improve the utility of our predictions by providing confidence intervals or prediction intervals. This can be done via simulation (for RF, using the distribution of trees outputs; for LSTM, maybe using dropout at prediction time to simulate ensemble, or training a quantile regression LSTM). Though this doesn't directly improve point accuracy, it provides more information. Specifically for volume, giving a range (e.g., there's a 95% chance volume will be between X and Y) might be valuable to risk management.

Interpreting Model Decisions: For greater trust and insight, one could use methods like SHAP (SHapley Additive exPlanations) or saliency maps for the neural nets to see which timesteps or features the model is relying on. This doesn't improve accuracy by itself, but could highlight weaknesses or suggest new features. For example, if a SHAP analysis on the RF showed that whenever last week's average volume is high it strongly pushes prediction up, then one could realize the model lacks a counter-feature to modulate that if some other condition is present.

6 Conclusion

In summary, our empirical evaluation shows that deep learning models (particularly LSTM) offer substantially better short-term volume forecasts than linear regression on cryptocurrency market data. The single-step LSTM achieved the highest accuracy (≈ 94% variance explained), while the sequence-to-sequence LSTM, GRU, Random Forest and CNN showed varying but lower performance. These findings are consistent with prior work reporting LSTM's superiority in financial time series and GRU's competitiveness. Notably, even multivariate models (using altcoin volume) yielded only marginal gains, likely due to strong collinearity.

Limitations and Future Work: The models still struggled with sudden volume surges, suggesting the need for additional inputs (e.g. volatility or order book features) or hybrid models. Incorporating exogenous signals (news, on-chain metrics) could further improve forecasts. Our study focused on 4-hour horizons; extending analysis to different frequencies (e.g. 1-hour, 1-day) or to probabilistic forecasting are possible extensions. Additionally, model ensembles or attention-based architectures might capture subtle patterns better.

Overall, our results provide a realistic benchmark for crypto volume forecasting and highlight that while deep RNNs significantly outperform simple baselines, forecasting accuracy remains limited by the market's intrinsic unpredictability.

References

[1] Latif, N., Selvam, J. D., Kapse, M., Sharma, V., & Mahajan, V. (2023). Comparative performance of LSTM and ARIMA for the short-term prediction of Bitcoin prices. Australasian Accounting, Business and Finance Journal, 17(1), 256–276.

[2] Wu, J., Zhang, X., Huang, F., Zhou, H., & Chandra, R. (2024). Review of deep learning models for crypto price prediction: implementation and evaluation. arXiv:2405.11431

[3] Gurgul, V., Lessmann, S., & Härdle, W. K. (2024). Deep learning and NLP in cryptocurrency forecasting: A dynamic bidirectional attention model. IEEE Access, 12, 188765–188778.

[4] Gupta, B. B., Gaurav, A., Piñeiro-Chousa, J., & López-Cabarcos, M. Á. (2025). Predicting the variation of decentralised finance cryptocurrency prices using deep learning and a BiLSTM-LSTM based approach. Enterprise Information Systems. (Forthcoming)

[5] McNally, S., Roche, J., & Caton, S. (2018). Predicting the price of Bitcoin using machine learning. In 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP) (pp. 339–343). IEEE.

[6] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.

[7] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32.

[8] Borovykh, A., Bohte, S., & Oosterlee, C. W. (2018). Conditional time series forecasting with convolutional neural networks. In Lecture Notes in Computer Science: Vol. 10827. Proceedings of the 24th International Conference on Neural Information Processing (ICONIP 2017) (pp. 538–554). Springer.

[9] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1724–1734).

[10] Siami-Namini, S., Tavakoli, N., & Siami Namin, A. (2018). A comparison of ARIMA and LSTM in forecasting time series. In 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 1394–1401). IEEE