Comparison of statistical and machine learning predictive models applied to time series : an application to the Bitcoin

Analyse & prévision de séries temporelles via des méthodes statistiques et de machine learning : cas d'étude sur le Bitcoin

Alek Tamoyan[1,2] and Bilal El Abdellaoui[1,2]

[1] Computer Science & Artificial Intelligence Department, École des Mines d'Alès, Alès, France
[2] Computer Science Department, České vysoké učení technické (ČVUT), Praha, Czech Republic

E-mail: alek.tamoyan@mines-ales.org & bilal.el-abdellaoui@mines-ales.org

Abstract

In the age of the digital revolution, crypto-currencies have emerged as intriguing and disruptive financial assets, generating unprecedented excitement among the economic community and investors alike. In this ever-changing context, the application of forecasting techniques (whether statistical or machine learning-based) is finding new relevance in predicting the value of crypto-currencies, such as the world-famous Bitcoin. This study delves into the potential of neural networks, especially the Long Short-Term Memory recurrent neural network (LSTM), in contrast to traditional time series models like the ARIMA and SARIMA. By estimating these models on the **Bitcoin stock market** index since its creation and comparing different performance indicators, we have ascertained that the Machine Learning method yields a significant reduction in prediction errors. These findings align with similar studies conducted on stocks from stock market indices and various financial assets.

Keywords: Bitcoin ; Machine Learning ; LSTM ; ARIMA ; SARIMA

Résumé

À l'ère de la révolution numérique, les cryptomonnaies ont émergé en tant qu'actifs financiers intrigants et disruptifs, suscitant un engouement sans précédent au sein de la communauté économique et des investisseurs. Dans ce contexte en constante évolution, l'application de techniques prévisionnelles (qu'elles soient à fondement statistiques ou bien de machine learning) trouve une nouvelle pertinence dans la prédiction des valeurs des crypto monnaies, à l'instar du célébrissime Bitcoin. Dans cette étude, nous explorons le potentiel d'un modèle utilisant les réseaux de neurones nommé les « Long-Short-Term-Memory » (LSTM) en comparaison avec les modèles de séries temporelles traditionnels tels que ARIMA & SARIMA. En entraînant ces modèles sur le cours du **Bitcoin** depuis sa création et en les évaluant au travers de différents indicateurs de performances, nous avons constaté que les LSTM permettent une réduction significative des erreurs de prédiction avec un temps d'entraînement optimal. Ces résultats concordent avec des études similaires menées sur des actions d'indices boursiers et sur divers actifs financiers.

1. Introduction

Analysing and predicting Bitcoin's trajectory carries substantial importance. In fact, Bitcoin stands as a pivotal digital asset, epitomizing the evolution of decentralized digital currencies. As a trailblazing cryptocurrency, its price movements wield substantial sway over investor sentiments and can decisively shape investment choices within the crypto realm and beyond. In fact, the maximum stock price of Bitcoin in 2021 reached over 64.000$ per unit, making a staggering increase of more than 900% in a year.

Moreover, Bitcoin's performance often serves as a *litmus test* for the overall health of the broader digital economy, particularly within the realm of blockchain technology and financial innovation. As blockchain's transformative potential reverberates across industries and economies, the behaviour of the Bitcoin index emerges as a pivotal indicator of market trends and economic prospects.

Coping with copious amounts of intricate financial data and unforeseen market developments amplifies the complexity. The non-linear characteristics of financial markets necessitate advanced forecasting models, yet inherent uncertainty underscores the paramount importance of risk management and diversification in effective investment strategies. Yet, as the volatility is high for most cryptocurrencies, the reward also might be high for the analysts that are able to compute great predictions.

Recent times have witnessed a paradigm shift in forecasting methodologies, driven by the integration of advanced machine learning and statistical techniques **[1].** Researchers and analysts are progressively harnessing data-driven models to predict stock prices. These models harness historical price data, trading volumes, and pertinent financial markers to formulate forecasts, thereby furnishing invaluable insights for investors and decision-makers.

Thus, traditional statistical methods have proven their worth **[1],** new machine learning methods leave some "surprises" and methods combining the two tend to appear [**2].** This paper compares the accuracy of different techniques (ARIMA, SARIMA, LSTM) when it comes to the forecast of Bitcoin prices. As the data collected is non-stationary, both methods ARIMA and SARIMA are used. The idea is to see whether traditional techniques still have their position on times series forecasting or not.

The article is organized as follows. The second section highlights what has been done in terms of time series forecasting. Section 3 dwells on the features & the mathematic behind the models. Section 4 underlines the data and the methodology used. The results will be provided and discussed on section 5.

2. Time Series Forecasting

Time series analysis and dynamic modelling constitute a captivating field of research with extensive practical applications across disciplines such as business, economics, finance, and computer science. The primary objective of time series analysis involves the examination of the sequential data points within a time series and the creation of a model that characterizes the underlying data structure. This, in turn, enables the prediction of future values within the time series.

Considering that time series forecasting methods rely on historical data to make predictions, they often lack robustness when confronted with the unpredictability of life or any significant events that can cause stock prices to either soar or plummet. For instance, a company's outstanding performance throughout the year could lead to such fluctuations. A change in the monetary policy could lead to a change in interest rates which can affect the stock market. These methods tend to exhibit greater reliability when applied to assets that demonstrate minimal price fluctuations in response to such events.

Given the critical role of accurate time series forecasting in various applied sciences, the development of effective forecasting models is imperative. Over time, a diverse array of time series forecasting models has emerged in the literature. Traditionally, time series forecasting, particularly in econometrics, has relied on ARMA models, a framework originally conceptualized by *Box and Jenkins* **[3].**

However, most time financial data such as stock prices present volatility to a degree that it becomes more complicated for the ARMA models [**4].** In fact, one of the most important hypotheses is the assumption of constant variance. Yet, an ARIMA model can be integrated with a Generalized Auto-regressive Conditional Heteroskedasticity (GARCH) model, but the optimization of an GARCH model might also be challenging **[5].**

Novel deep learning techniques have emerged to tackle the challenges associated with forecasting models. One such technique is LSTM (Long Short-Term Memory), a specialized variant of Recurrent Neural Networks (RNNs) introduced by *Hochreiter and Schmidhuber* **[6].** Despite being a relatively new approach for addressing prediction problems, deep learning-based methodologies have garnered considerable attention among researchers. *Lee and Yoo* introduced an RNN-based approach to predict stock returns, which involved constructing portfolios by adjusting return threshold levels using the internal layers of the RNN **[7].**

Methods like ANN, RNN and LSTM were also compared in the forecasting of Bitcoin prices [**8].** *Siami-Namini*, *Tavakoli* and *Siami-Niamin* compared the ARIMA and LSTM models through different stock prices and concluded that the LSTM model improved the prediction by 85% on average **[9].**

In this paper, a comparative analysis is conducted on the performance of both ARIMA and SARIMA models against the LSTM model in the prediction of Bitcoin prices.

3. Mathematics behind the models

3.1 ARIMA & SARIMA models

ARIMA, which stands for Auto Regressive Integrated Moving Average, is a composite time series modelling approach that combines the elements of autoregressive (AR) processes and moving average (MA) processes.

The "AR" process describes how each current value denoted X_t depends on its previous p values, weighted by the autoregressive coefficients. One equation corresponding to this model is :

$$\mathbf{x_t = \phi_1 X_{t-1} + \dots \phi_p X_{t-p} + c + \varepsilon_t}$$

x_t is the current value, c is constant, the terms ϕ_i are the autoregressive coefficients at different lags and ε_t represents the white noise error term at time t.

Thus, the order p of the model determines the number of lags considered, with higher orders capturing more complex dependencies within the time series data.

The "MA" process takes into account the dependency between observations and the residual error terms when a moving average model is used to the lagged observations. An MA model of order q can be written as :

$$\mathbf{x_t = \mu + \theta_1 \varepsilon_{t-1} + \dots + \theta_p \varepsilon_{t-p}}$$

μ is the expectation of X_t (often assumed equal to zero), the θ_i terms are the weights applied to the different values of ε_i. We make the assumption that ε_t follows a Gaussian white noise process characterized by a mean of zero and a variance of $\sigma\varepsilon^2$.

By combining these two models, we can obtain an ARMA model with parameters p and q each characterizing the AR and MA process :

$$\mathbf{x_t = \Sigma_{i = 1, p}\, \phi_i X_{t-i} + \varepsilon_t + \Sigma_{i = 1, q}\, \phi_i X_{t-i}}$$

The I that appears in ARIMA stands for *Integrated*. This step involves differencing the time series data to make it stationary by measuring differences between observations. This step enables us to convert a non-stationary time series into a stationary one. To perform a first order difference you subtract each observation from the previous one. Mathematically, for a time series X_t, the first difference is calculated as :

$$\mathbf{dX_t = X_t - X_{t-1}}$$

Eventually, the acronym ARIMA(p, d, q) represents the essential components of this model.

When working with time series data that exhibit seasonal patterns, it's likely that short-term factors unrelated to the seasonality affect the model. Therefore, there's a requirement to construct a seasonal ARIMA model that incorporates both the non-seasonal and seasonal aspects using a multiplicative framework. The typical specification for a seasonal ARIMA model is represented as ARIMA(p, d, q) × (P, D, Q,S) where:

- "p" denotes the order of non-seasonal autoregressive (AR) components.
- "d" represents the degree of non-seasonal differencing.
- "q" stands for the order of non-seasonal moving average (MA) components.

- "P" signifies the order of seasonal autoregressive (AR) components.
- "D" represents the degree of seasonal differencing.
- "Q" indicates the order of seasonal moving average (MA) components.
- "S" corresponds to the length of the repeating seasonal pattern.

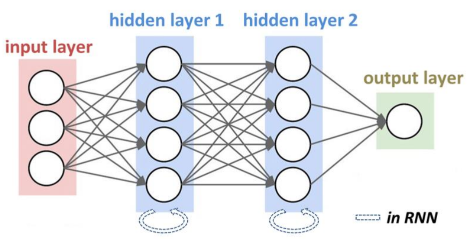
To estimate a seasonal ARIMA model, we need to find the appropriate values for (p, d, q) and (P, D, Q).

To start this process, we first inspect the data's time plot. For instance, if the data's variance shows an increasing trend over time, we should consider using methods to stabilize variance and apply differencing techniques but also be cautious about excessive differencing.

Next, we utilize the autocorrelation function (ACF) to assess the degree of linear dependence between observations separated by a lag of p in the time series. Concurrently, we use the partial autocorrelation function (PACF) to establish the necessary number of autoregressive terms (q). By employing these diagnostic tools, we can determine initial values for the autoregressive order (p), differencing order (d), moving average order (q), and their corresponding seasonal counterparts (P, D, Q).

3.2 Recurrent Neuron Network (RNN) & LSTM

RNNs are an extension of neural networks, designed and adapted to process sequential data, such as time series. Unlike classical neural networks, RNNs have a loop structure, where the previous output is fed back into the network at the next step. This enables them to maintain an internal "memory" and take into account past contextual information when processing data.



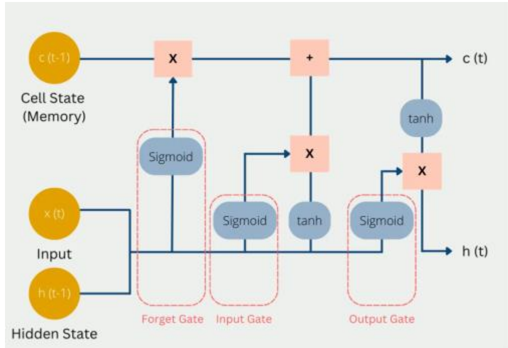
(Fig.1) RNN Architecture **[15]**

(Fig.1) presents an example of the interconnections between diverse layers and neurons within a recurrent neural network. This representation highlights the fact that the output extracted from one layer assumes the role of an input for neurons residing in prior layers. These neurons, collectively constituting recurring units, undertake computations during each time step to create an output (y_t). Through the time, the neuron acquires a novel input vector (x_t) while adding the previously output (y_{t-1}) into this vector — an entity referred to as the "recurrent input." Thus, the neuron conducts computations of the output vector using the input vector (x_t) alongside the recurrent input (y_{t-1}) facilitated by an activation function called σ.

Considering W_x as the weight matrix that multiply the input vector and W_y the weight matrix that multiply the recurrent vector, y_t can be expressed this way :

$$y_t = \sigma(W_x * x_t + W_y * y_{t-1} + b_y) \textit{ with b a bias vector}$$

However, conventional RNNs often suffer from the "Vanishing Gradient Problem" **[10]** when learning long sequences. In order to deal with this issue, which occurs when gradients become extremely small as they propagate through the time, we use LSTMs. Proposed by *Hochreiter and Schmidhuber* in 1997 **[6]** , Long-Short Term Memory network introduces a more complex internal structure called a "memory cell", which is capable of storing and updating information over long periods of time. They excel in propagating activations over extensive periods, handling sequences with distant dependencies . LSTMs effectively address the vanishing gradient problem by introducing blocks that modify the recurrent unit. These blocks include an additional cell and multiple gates that regulate information flow within the unit. The key components of an LSTM cell are pictured in (Fig.2) :



(Fig.2) LSTM Architecture **[16]**

The cell state (c_t) is functioning as a repository of long-term memory, the cell state preserves information over sequences.

The forget gate determines which elements from the previous cell state (c_{t-1}) should be discarded. It takes into account the previous hidden state (h_{t-1}) and current input (x_t).

The input gate governs the assimilation of new information into the cell state. It relies on the current input (x_t) and prior hidden state (h_{t-1}).

The output gate is responsible for the information emitted as the ensuing hidden state (h_t), the output gate integrates the present input (x_t) and adapted cell state (c_t).

Thus, the mathematical expression of LSTM operations is expressed as :

$$f_t = \sigma\left(W_{xf} * x_t + W_{hf} h_{t-1} + b_f\right)$$

$$i_t = \sigma(W_{xi} * x_t + W_{hi} h_{t-1} + b_i)$$

$$o_t = \sigma(W_{xo} * x_t + W_{ho} h_{t-1} + b_o)$$

$$C_{t_{bis}} = tanh\,(W_{xc} * x_t + W_{hc} h_{t-1} + b_c)$$

$$C_t = f_t \circ C_{t-1} + i_t \circ C_{t_{bis}}$$

$$h_t = o_t \circ tanh\,(C_t)$$

3.3 Evaluation Metrics

In this study, the model performance evaluation is done by comparing the predicted values with their corresponding observed values typical performance metrics. We choose to pursue with 3 well-known of them which are the **MAE** (Mean Absolute Error), the **MSE** (Mean Squared Error) and the **MAPE** (Mean Absolute Percentage Error). The MAE criteria simply penalizes the extent to which the predicted value differs from the actual value. The MSE criteria also penalizes but accentuates larger errors (due to squaring). Formulas are presented in Table 1.

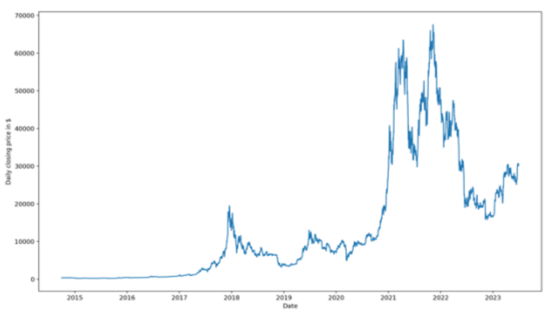
Metric	Formula
MSE	$\frac{1}{N}\sum_{i=1}^{N}(y^i_{prediction} - y^i_{real})^2$
MAE	$\frac{1}{N}\sum_{i=1}^{N}\lvert y^i_{prediction} - y^i_{real}\rvert$
MAPE	$\frac{1}{N}\sum_{i=1}^{N}\left\lvert \frac{y^i_{prediction} - y^i_{real}}{y^i_{real}}\right\rvert$

(Table 1.) Model Performance Metrics

4. Datasets and Methodology

4.1 Dataset

Thanks to the module "yFinance", the historical data of Bitcoin were extracted. Those historical data contained daily Bitcoin information with different features : Open, High, Low, Close, Adjusted close and Volume. We choose to focus our experimental study on the closing price of Bitcoin from 01 Oct 2014 to 01 July 2023 which corresponds to 3195 observations. The time series was split into two subsets : training and validations datasets where 80% of the dataset was used for training and the remaining 20% for validation.
TC prices and the specific trends or seasonality if any. As disclosed before, the dataset contains the daily closing price of BTC throughout years.



(Fig 3.) Bitcoin stock prices

As shown in (Fig 3.), the Bitcoin prices appeared to be nearly constant over the first three years of our plot and then have increased a lot with high volatility in the recent years.

4.2 Methodology

4.2.1 - ARIMA & SARIMA

In all the statistical approaches considered, the requirement for obtaining a stationary time series has been addressed by determining the necessary degree of differencing. Just as observed in numerous economic time series studies **[11]**, achieving stationarity necessitated only a single integration.

In (Fig 3.), it is difficult to detect any kind of seasonality, but we can already assumes that the series is non-stationary. Thus, to perform ARIMA & SARIMA the series needs to be stationary. Using the log function was not sufficient to make it stationary based on the Augmented-Dickey-Fuller Test differencing was used with a degree of 1.

As to perform ARIMA, the right parameters (p,d,q) needed to be found. As a first order differencing is used and is sufficient, the parameter d is set to 1.

In order to find the parameters p and q, an usual method is to examine the Autocorrelation Function (**ACF**) and the Partial Autocorrelation Function (**PACF**). Different combinations were tested through this examination, we then used an algorithm that is designed to automate the process of selecting the best ARIMA model for a given time series data set by trying different combinations of parameters (d still being set to 1) and selecting the ones that minimize a chosen criterion.

In the case of ARIMA & SARIMA, the selection of appropriate parameters has been executed through the utilization of AIC (Aikake Information Criterion) criteria.

As to perform SARIMA, we analysed the seasonality of the time series by plotting multiple graphs based on daily, weekly, monthly and quarterly prices. Different combinations were then explored before using an algorithm that chooses the best ones.

4.2.2 – LSTM

For the LSTM Neural Network model, standardization has been applied to the series for both training and testing phases. Numerous studies have affirmed the enhancement of the learning process in neural networks due to standardization **[12][13].** Thus, "MinMaxScaler" library was used for standardization, while Adam optimizer was employed for optimization of the model.

5. Results & discussion

5.1 Results

This study aims to validate the effectiveness of machine learning models in contrast to traditional models used for time series analysis. Specifically, we will conduct a comparative analysis with the "ARIMA" family methods, encompassing ARIMA and SARIMA. For this purpose, each of them has been applied to the bitcoin series with daily data starting from October 2014. This choice of date was made for convenience concerning seasonality.

Hardware & Software	Specification
Memory	8 GB
Processor	2020 M1 Ship (8-Core CPU)
GPU	2020 M1 Ship (8-core GPU)
Operating System	macOS
Deep Learning Library	Tensorflow
Programming Language	Python

(Table 2.) List of Hardware & Softwares used

Our LSTM-based model consists of three stacked layers followed by a dropout layer with a 0.2 fixed dropout rate. After these LSTM layers, a dense layer is employed to process the learned features. The time window, representing the historical information considered for each prediction, has been established at 90 days for seasonality purposes.

This architectural configuration has been designed to extract and capture intricate temporal dependencies of the Bitcoin time series and has been shaped through time with experience. The LSTM Model's summary is displayed in (Table 3.) next to the computing device used to run the program (Table 2.). Finally, the results of the different evaluation metrics are displayed in a summary (Table 4.)

Layers	Parameters
Layer 1	128
Layer 2	128
Layer 3	64
Dropout	0.2
Dense	1

(Table 3.) LSTM Model's summary

Evaluation Metrics	ARIMA	SARIMA	LSTM
MSE	$2{,}42.10^{6}$	$1{,}86.10^{6}$	$4{,}84.10^{5}$
MAE	1384.6	1155.2	454.8
MAPE	0.050	0.041	0.013

(Table 4.) Evaluation Metrics for our different models

5.2 Discussion

As shown in (Table 4.), even though SARIMA performed better than ARIMA the LSTM model appears to be the best model here. Based on the MAE, a use of a LSTM model enables in this case a reduction of 67,24 % in the metric for ARIMA and 60,52 % for SARIMA.

Here the comparison was made on Bitcoin, an asset known to be very volatile **[16],** and not following a classical time series scheme with a certain seasonality and a clear trend or dependencies between close values. Moreover, this asset has shown itself to be relatively fragile in the face of market news and sentiment.

Its value is often influenced by external factors and breaking news. Sudden regulatory announcements, market-moving tweets from celebrities, or shifts in investor sentiment can lead to rapid and dramatic price fluctuations. This lack of robustness underscores the need for caution when realizing such predictions only based on historical data.

In that case, the LSTM model has a kind of advantage in case of prediction due to its architecture that us capable of capturing more complex data compared to the more classical schemes of ARIMA and SARIMA. But the ARIMA model still was expected to perform lower as the LSTM one **[9].** The idea of this paper was to go further by having this comparison with SARIMA as well and making it on a different kind of asset (cryptocurrencies).

Eventually, traditional techniques should be preferred when it comes to more classical data, for instance the number of sell of an e-commerce website or the usual case of the number of passengers of an airplane company. On top of that, the training of those two models (ARIMA & SARIMA) took significantly less time than the training of the LSTM one.

6. Conclusion

This study delves into an analysis of the Long Short-Term Recurrent Neural Network's (LSTM) efficiency when compared to classical time series models. The primary finding underscores a significant reduction in prediction error using ML methods. This outcome resonates with prior research conducted on different financial assets. This highlights the potential of recurrent neural networks, particularly LSTM, as a viable option for constructing models that predict cryptocurrencies.

In other ways, different other searchers have proposed models that seem to be very efficient alternatives. In the ML area, the N-BEATS (Neural Basis Expansion Analysis for Time Series) model, proposed by Boris Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio, is a powerful neural network architecture tailored for time series forecasting tasks **[14]**. It employs stacked blocks to capture both short-term and long-term patterns in data, which include trend and seasonality components. Notably, N-BEATS stands out with its ability to efficiently parallelize training, leading to rapid convergence. It achieves state-of-the-art forecasting accuracy, frequently surpassing traditional methods and other neural network models. With the capability to decompose time series into interpretable basis functions, its flexibility in accommodating various data types, and its impressive performance, N-BEATS showcases efficiency by achieving more than competitive forecast accuracy across diverse time series scenarios.

References

[1] Salvail, A. (2021). Sur l'utilisation de l'apprentissage profond pour la prévision probabiliste en séries chronologiques. Faculté des sciences, Université de Sherbrooke.

[2] TY - BOOK, AU - Oreshkin, Boris, AU - Carpo, Dmitri, AU - Chapados, Nicolas, AU - Bengio, Yoshua, PY - 2019/05/24 - N-BEATS: Neural basis expansion analysis for interpretable time series forecasting

[3] G. Box, G. Jenkins, *Time Series Analysis: Forecasting and Control*, San Francisco: Holden-Day, 1970.

[4] Andrea-Cristina PETRICĂ & Stelian STANCU & Alexandru TINDECHE, 2016. "Limitation of ARIMA models in financial and monetary economics," Theoretical and Applied Economics, Asociatia Generala a Economistilor din Romania - AGER, vol. 0(4(609), W), pages 19-42, Winter

[5] Kane, M.J., Price, N., Scotch, M. et al. Comparison of ARIMA and Random Forest time series models for prediction of avian influenza H5N1 outbreaks. BMC Bioinformatics 15, 276 (2014). https://doi.org/10.1186/1471-2105-15-276

[6] S. Hochreiter, J. Schmidhuber, "Long Short-Term Memory," *Neural Computation* 9(8):1735-1780, 1997.

[7] S. I. Lee, S. J. Seong Joon Yoo, "A Deep Efficient Frontier Method for Optimal Investments," Department of Computer Engineering, Sejong University, Seoul, 05006, Republic of Korea, 2017.

[8] Dr. G. Hegde, J. Arunan, F. Modak, A. Nair, K. Sangal, Bitcoin Price Prediction using Deep Learning. Department of Computer Engineering, Pillai College of Engineering, Mumbai, India, 2023

[9] A Comparison of ARIMA and LSTM in Forecasting Time Series. / Siami-Namini, Sima; Tavakoli, Neda; Siami Namin, Akbar. Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018. ed. / M. Arif Wani; Moamar Sayed-Mouchaweh; Edwin Lughofer; Joao Gama; Mehmed Kantardzic. Institute of Electrical and Electronics Engineers Inc., 2019. p. 1394-1401 8614252 (Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018).

[10] Hochreiter, Sepp. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 6. 107-116. 10.1142/S0218488598000094.

[11] McCabe,B.P.M. & Tremayne,A.R., 1995. "Testing a Time-Series for Difference Stationarity," Cambridge Working Papers in Economics 9420, Faculty of Economics, University of Cambridge. <https://ideas.repec.org/p/cam/camdae/9420.html>

[12] Zhang, G., Patuwo, B.E. and Hu, M.Y. (1998) Forecasting with Artificial Neural Networks: The State of the Art. International Journal of Forecasting, 14, 35-62. http://dx.doi.org/10.1016/S0169-2070(97)00044-7

[13] Lachtermacher, G. and Fuller, J.D. (1995), Back propagation in time-series forecasting. J. Forecast., 14: 381-393. https://doi.org/10.1002/for.3980140405

[14] Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2020). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting.

[15] Schema made by : https://www.researchgate.net/figure/Generalized-recurrent-neural-network-architecture-with-two-hidden-layers-The-NN_fig3_335159004

[16] Schema made by https://databasecamp.de/en/ml/lstms (based on https://towardsdatascience.com/lstm-recurrent-neural-networks-how-to-teach-a-network-to-remember-the-past-55e54c2ff22e)

[17] Gupta H, Chaudhary R. An Empirical Study of Volatility in Cryptocurrency Market. *Journal of Risk and Financial Management*. 2022; 15(11):513. https://doi.org/10.3390/jrfm15110513