Soft Computing (2024) 28:12827–12834
https://doi.org/10.1007/s00500-024-10301-4

APPLICATION OF SOFT COMPUTING

Bitcoin price prediction using LSTM autoencoder regularized by false nearest neighbor loss

Chahat Raj[1] · **Manojit Chattopadhyay**[2]

Accepted: 27 August 2024 / Published online: 19 November 2024

Abstract
We implement deep learning for predicting bitcoin closing prices. Identifying two new determiners, we propose a novel LSTM Autoencoder using Mean Squared Error (MSE) loss which is regularized by False Nearest Neighbor (FNN) algorithm. The method results in reduced error rates when compared to traditional forecasting algorithms and is statistically validated. This research contributes by developing a robust algorithm that accurately determines the fluctuation directions in bitcoin prices and results in values closer to the actual prices.

Keywords Cryptocurrencies · Bitcoin · Time series forecasting · LSTM · Autoencoder · False nearest neighbor · Regularizer · Deep learning

JEL Classification G1 · C6 · C12

1 Introduction

Deep learning techniques brought a massive revolution in predicting stock prices, equities, mutual funds, gold, and silver, amongst other financial instruments. Nonlinearity and high volatility made the prediction difficult for financial time series (Chen et al. 2020; Chen et al. 2024; Lux 2013; Madan 2015; Segnon & Bekiros, 2020) and price time series (Mari and Mari 2021). Antulov-Fantulin et al. (2021) proposed intraday volume prediction of bitcoin using temporal mixture ensemble model. The trend is shifting towards predicting prices of cryptocurrencies, especially Bitcoin (Nakamoto 2008), which has gained popularity after a multi-fold surge in its worth in a short span of six months, a phenomenon that rarely occurs in financial markets (Gandal et al. 2021). Bitcoin exhibits diverse challenges and opportunities as well as various similarities with gold (Ghorbel and Jeribi 2021). These fluctuations have severely raised the necessity of bitcoin price predictions to mitigate market risks like bitcoin trend prediction (Cavalli and Amoretti 2021; Figa-Talamanca and Patacca 2019). The bitcoin price behaviour has studied detail in the work of Cretarola et al. (2020).

In the related domain of finance few Deep learning based Long Short-Term Memory (LSTM) algorithm has been known to use for successful prediction task (McNally et al. 2018; Maknickien and Maknickas, 2012; Ahmed et al. 2020; Nelson et al. 2017; Rezaei et al. 2021; Na and Kim 2021). In stock market forecasting, traditional statistical and artificial intelligence methods are mainly applied that involved single value prediction. In different time horizons, trading strategy has been elaborated using LSTM for predicting bitcoin price (De Angelis et al. 2021). In contrast, the latest deep neural network models are applied to develop models for multiple inputs and multiple outputs based on the LSTM network (Ding and Qin 2020). However, accurate real-time market predictions have not yet been achieved because of financial markets' volatility and chaotic nature.

Liu et al. (2021) investigated various determiners impacting bitcoin prices. However, the impact of new-age determiners such as behavioral investment and the number of blockchain wallet users has not been discussed. Bitcoin prices remained steady below a threshold of USD 15,000

✉ Manojit Chattopadhyay
mchattopadhyay@iimraipur.ac.in

Chahat Raj
craj@gmu.edu

[1] Department of Computer Science, George Mason University, Fairfax, USA

[2] Information Systems, Indian Institute of Management Raipur, Chhattisgarh, India

until October 2020, even when the number of blockchain wallet users was increasing quadratically. From October 2020 till April 2021, the latter's value rose exponentially, while a multi-fold increase in bitcoin's price was observed (https://www.statista.com/chart/24793/selected-cryptocurrency-price-growth/). This relationship is displayed in Fig. 1. Studies show that every increase in bitcoin price is more likely to generate a two-fold increase in bitcoin price than observing a decrease (Phillips et al., 2018). This helps us understand the increase in blockchain wallet users as the bitcoin price increased, followed by a saturation (no significant increase) in the growth of blockchain wallet users that caused the price to fall from a peak of USD 60,000 to USD 35,000 during April 2021 to May 2021 (https://www.statista.com/statistics/647374/worldwide-blockchain-wallet-users/). This rapid investment and disinvestment pattern, which we term as behavioral investment, has a two-way relationship with high short-term fluctuations in the price of cryptocurrencies such as bitcoin. We identify this relationship and establish two new fluctuation determiners in this paper: behavioral investment and the number of blockchain wallet users.

To improve forecasting performance arising out of such fluctuations, we utilize a price convergence method incorporating the False Nearest Neighbor (FNN) algorithm introduced by Kennel et al. (1992). Gilpin (2020) adapted this method on time-series data to reconstruct strange attractors. The idea of the proposed method is derived from the fact that in an unpredictable environment such as the financial markets, deterministic conditions are non-linear and largely scattered in nature. The considerable factors are complex, a large portion of which is outliers, thus making the projection noisy. These multi-dimensional representations are linearly presented using Delay Coordinate Embedding (DCE). The necessary condition for the technique implies that the number of dimensions is known, which is not a plausible condition in financial applications. This is when FNN's use becomes indispensable, where the embedding dimension of time-series data is determined, thus overcoming the above-stated limitation of DCE.

This paper presents an LSTM autoencoder model that uses FNN as a regularizer to a traditional loss function. The method bridges a necessary gap in the existing research by bringing the bitcoin fluctuation deterministic factors into linear space, reducing noisy outliers. The deep learning method considers the critical real-world factors, where the neurons in the network automatically perform feature extraction. The use of LSTM in our approach is supported by the fact that it offers highly retentive memory with the use of forget gates. This allows the network to remember necessary factors for a long time and selectively ignore unnecessary information. Autoencoder's use is eminent for the reconstruction of information in time-series data. Overall, the method considers selectively necessary information from a multi-dimensional space (financial market factors), reconstructs them using an autoencoder, and performs prediction by exploiting the memory power of LSTMs. This justifies the importance of the proposed method. This novel method improves prediction accuracy compared to traditional forecasting algorithms, with a reduced error rate and supplements precision in bitcoin price prediction.

2 Data and methodology

The dataset used in this research for bitcoin price prediction is procured from the coindesk website (Coindesk 2021), similar to Demir et al. (2018), Dyhrberg (2016), and Katsiampa (2017). This dataset contains 24-hour bitcoin prices for 2791 days (1/10/2013 to 23/5/2021). We utilize the daily bitcoin closing prices to forecast future prices.

Predictions are calculated for immediate next-day bitcoin prices to several days' forecasts by altering the timestep count. The prices are scaled into a range of [0,1] to minimize

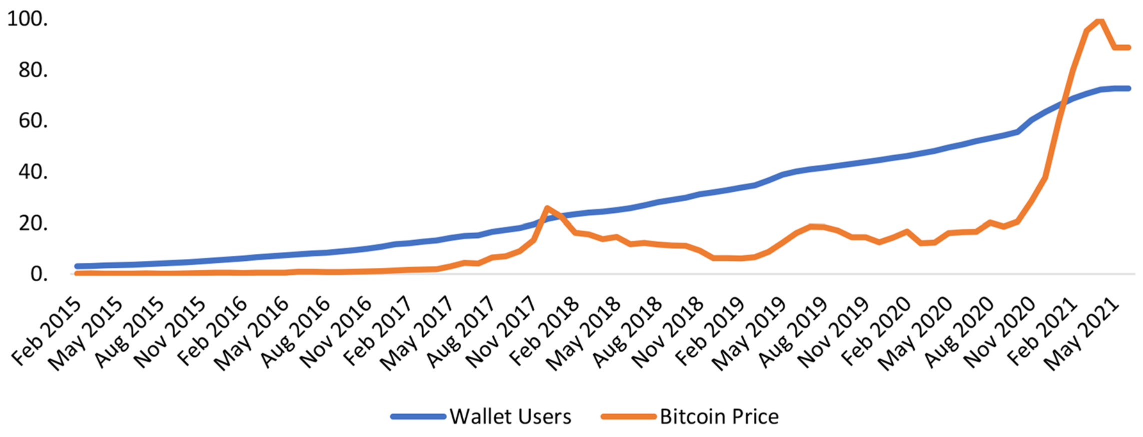

Fig. 1 Relationship between bitcoin price and number of blockchain wallet users

the magnitude impact of significant figures. Bitcoin closing price trend for 2791 observations is represented in Fig. 2.

2.1 LSTM autoencoder

Recurrent Neural networks (RNN) are primarily used in sequential data applications. LSTM is prevalent in this domain as it offers feedback connections in addition to feedforward transmissions. It is built as a self-supervised learning algorithm where it can learn from the previous sequential inputs. It can retain and store information across large data sequences possessing the ability to remember past patterns. This makes it efficient for prediction tasks. An autoencoder is a neural network containing a single hidden layer capable of learning from compressed input representations. It performs compression of information until the model's midpoint and reconstructs it back into the input data. The input vector $x \in [0{,}1]_d$ with dimension, d is mapped into a hidden vector $y \in [0{,}1]_d$. Here y is the compressed input representation at the midpoint bottleneck. The layers performing this compression constitute the encoder $f_\varnothing$. The mapped representation y is then mapped back into the input space as a vector $z \in [0{,}1]_d$ using a decoder $g_{\varnothing'}$. The vectors y and z are calculated using Eq. (1) and Eq. (2).

$$y = f_\varnothing(x) = s(W_x + b) \tag{1}$$

$$z = g_{\varnothing'}(y) = s(W'_y + b') \tag{2}$$

The error encountered during the reconstruction phase is reduced by optimizing $\varnothing$ and $\varnothing'$ using a loss function L as Eq. (3).

$$\begin{aligned}\emptyset, \emptyset' &= \underset{\emptyset,\emptyset'}{arg\,min} \frac{1}{n} \sum {}_{i=1}^{n} L(x^{(i)}, z^{(i)}) \\ &= \underset{\emptyset,\emptyset'}{arg\,min} \frac{1}{n} \sum {}_{i=1}^{n} L(x^{(i)}, g_\emptyset(f_\emptyset\left(x^{(i)}\right)))\end{aligned} \tag{3}$$

2.2 False nearest neighbor regularizer

Suppose in a d-dimensional space, neighbors of a point on a trajectory are close enough, supposedly overlapping, but are largely separated in a d+1 dimensional space. In that case, these are referred to as false neighbors. Such points co-exist only in a d-dimensional space. According to the traditional FNN algorithm, the correct embedding dimension d_E is obtained when FNN converges to zero as d is increased. In an m-dimensional space, to calculate the false neighbor count F_m, the input batch-size, B and the number of nearest neighbors, K are treated as hyperparameters as in Eq. (4). K is chosen upon deciding the optimum number of neighbors overlapping or close enough to be sufficiently informative. K is dependent on B setting $K = \max(1, ?0.01B?)$.

$$F_m = \frac{1}{KB} \sum {}_{k=1}^{K} \sum {}_{b=1}^{B} F_{kbm} \tag{4}$$

The false nearest neighbor loss, L_{FNN}, is then calculated using F_m, and the batch activations $h \in R^{B\times L}$ of a latent layer with L units as in Eq. (5).

$$L_{FNN} = \sum {}_{m=2}^{L} (1 - F_m) h_m^2 \tag{5}$$

Instead of using a traditional evaluation metric, like the MSE, to calculate reconstruction loss, we add the FNN loss with an adequate weightage λ to the standard metrics. This λ acts as a regularizer and brings the total loss to converge to zero, thus increasing the prediction accuracy. The loss function is given by Eq. (6), in which $\frac{1}{n} \sum {}_{i=1}^{n} (Y_i - Y_i)^2$ denotes the MSE, and λ takes variable values between 0 and 1. The weightage λ differs for each application depending upon the dataset and is optimized by iterative experiments.

$$Loss = \frac{1}{n} \sum {}_{i=1}^{n} (Y_i - ?_i)^2 + \lambda\, L_{FNN} \tag{6}$$

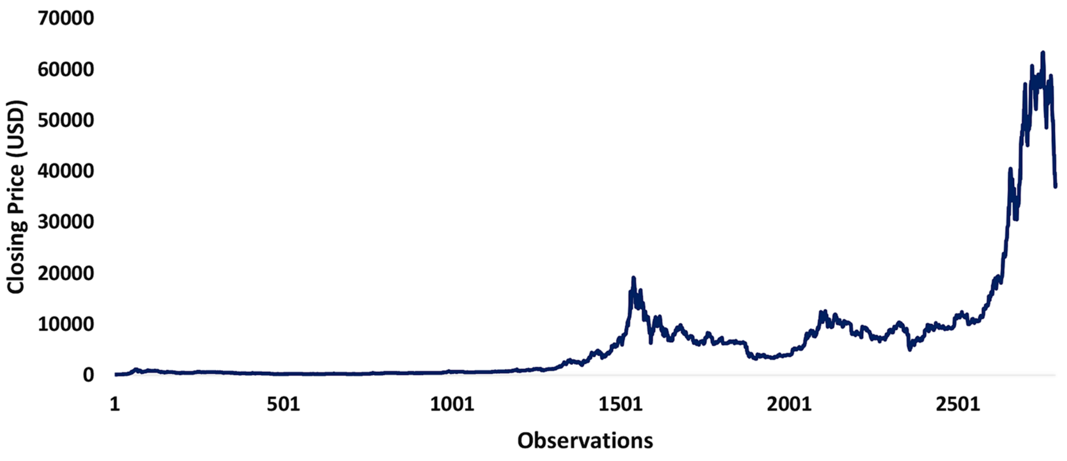

Fig. 2 Bitcoin price trend

The traditional False Nearest Neighbor (FNN) algorithm, outlined in Eq. (4), has notable limitations. It assumes that the algorithm's convergence to zero as the embedding dimension increases is consistent with the actual data structure, which may not always be the case (Kennel et al. 1992). Key hyperparameters, such as batch size B and the number of nearest neighbors K, are critical and often complex, with K depending on B, complicating parameter tuning (Fraser and Swinney 1986). This sensitivity and the dual summation over neighbors and batches can cause inefficiencies and inconsistencies, particularly with large datasets (Sugihara and May 1990; Grassberger & Procaccia 1983).

Equation (5) further illustrates limitations, as the FNN algorithm assumes data points exist in specific dimensions, which may not always match the actual data structure (Kennel et al. 1992). Finding the correct number of dimensions can be slow and inconsistent (Abarbanel et al. 1993). Precise settings for batch size and the number of nearest neighbors are essential, and incorrect settings may lead to inaccuracies (Fraser and Swinney 1986). The dependency on batch size complicates the tuning process, as adjustments require re-tuning of parameters (Cao et al. 1997).

Equation (6) relates to the computation of the loss function L_{FNN}, which is complex and involves multiple steps, making it difficult to implement and understand (Kennel et al. 1992). Accurate hyperparameter settings are crucial; incorrect settings can impair performance (Fraser and Swinney 1986). Integrating FNN loss with standard metrics increases computational demands, slowing down training (Abarbanel et al. 1993). The introduction of a weighting factor (λ) adds complexity, requiring extensive tuning (Cao et al. 1997). The algorithm's sensitivity to noise and the need for substantial model adjustments limit its accessibility and effectiveness (Kantz and Schreiber 2004; Sauer et al. 1991).

2.3 Proposed architecture

We employ an LSTM autoencoder network regularized with FNN loss to generate bitcoin price forecasts. The neurons in this deep learning model consider the effect of determinants responsible for bitcoin price change. The model selectively chooses information highly influencing the price change and discards useless information. The architectural representation of the proposed network is provided in Fig. 3. It consists of two modules: an encoder and a decoder. The encoder takes an input array in shape (n_input $\times$ features $\times$ timesteps). The number of features is set to be 1. The input is fed to an LSTM layer with 256 units. The output is fed to another LSTM layer that reduces the output feature size to 128. Next, we add a Dropout layer with a probability of 0.2 that solves the problem of overfitting. This output is the compressed feature vector of the fed input. The next layer in the sequence is a RepeatVector layer that is used for feature vector replication. This layer acts as a connection between the encoder and the decoder. In the decoder module, two LSTM layers are added in the sequence opposite the encoder. The first LSTM layer contains 128 units, and the second layer comprises 256 units. Another Dropout layer of probability 0.2 is added. The final layer is a TimeDistributed Dense layer with units equal to the number of features = 1.

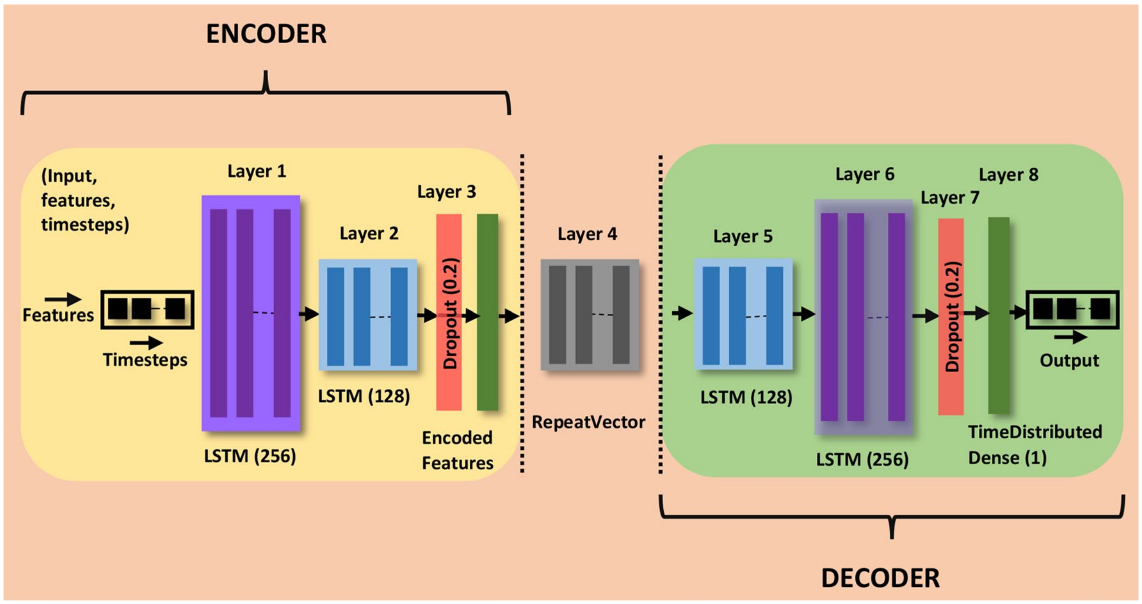

Fig. 3 Proposed LSTM autoencoder architecture

Table 1 Parameters and their values for experiments

Parameters	Value
Training Split	0.8
Batch Size (B)	64
Neighbors (K)	1
Latent layer units (L)	10
FNN regularizer weight (λ)	0.3
Epochs	100
Activation Function	ReLU
Optimizer	Adam
Language	Python 3
IDE	Google Colab

Table 2 RMSE scores for different number of prediction days

Horizon (days)	RMSE
1	0.5782
2	0.5788
3	0.5834
4	0.5893
5	0.5893
10	0.6352
20	0.6353
30	0.6472
60	0.6491

3 Numerical experimentation

All experiments are carried out using Python 3 on Google Colab. The data points are normalized using sklearn's StandardScaler. We split the dataset with an 8:2 ratio into training and testing sets. The value for timesteps is variable for each experiment and depends on the number of days we wish to predict prices. The number of units in the latent layer, L , is set to 10 as proposed by the existing studies. The loss function for LSTM Autoencoder is the combination of MSE and FNN loss with λ as the adequate scaler weight set to 0.3. Rectified Linear Unit (ReLU) is used as the activation function for LSTM layers in both the encoder and decoder. The model uses an Adam optimizer which keeps its learning rate variable throughout the training phase. The batch size is 64, and the model is trained for 100 epochs. Table 1 lists out the values of all parameters used.

3.1 Results and discussions

Root Mean Square Error (RMSE) is a widely accepted measure for determining time-series prediction accuracies. The error is calculated by the difference between the actual bitcoin prices and the predicted prices. RMSE, given by Eq. (7), is the average of these error values.

$$RMSE = \sqrt{\frac{1}{N}\sum (Y_i - ?_i)^2} \tag{7}$$

The empirical results demonstrate that the proposed method is an optimum model for bitcoin price prediction. The prediction error values for varied timesteps are listed in Table 2. Figure 4 plots the bitcoin price trend for actual and predicted values. The training and testing sets contain 2232 and 559 observations, respectively. It is observable that the proposed method efficiently adapts to the price fluctuations and predicts the highs and lows of the market quite well. The curve for predicted values neatly overlaps the actual price curve during the training phase and shows minor deflections during the testing phase.

An ablation study is performed by removing the FNN loss component to highlight the effect and usefulness of the regularizer. Figure 5 depicts the ablation comparison graphically for the testing phase. It is observed that while the LSTM Autoencoder with only MSE loss can correctly predict the fluctuating directions, there is a vast gap between the curves that appears to be constant along the trajectory. This prediction gap can be treated as a constant c which the

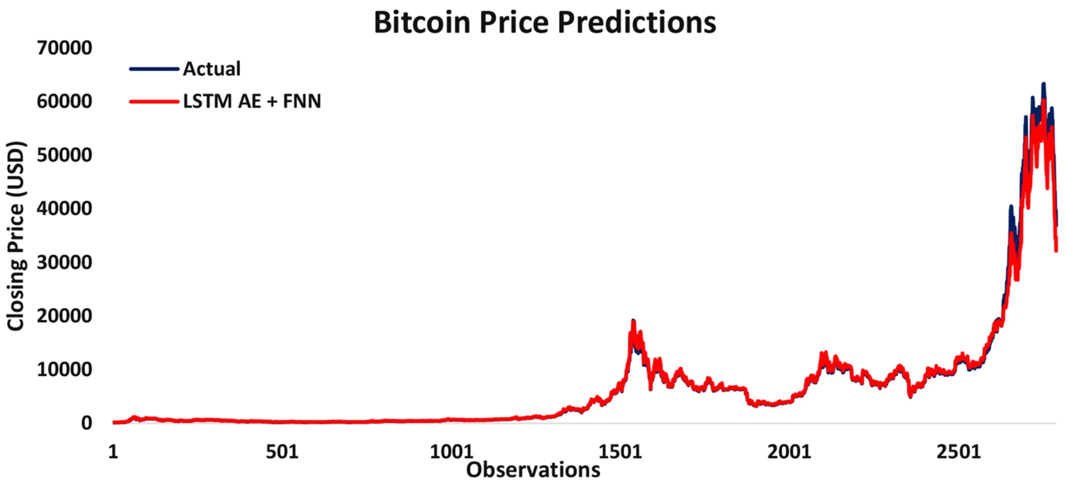

Fig. 4 Actual and predicted values of bitcoin prices

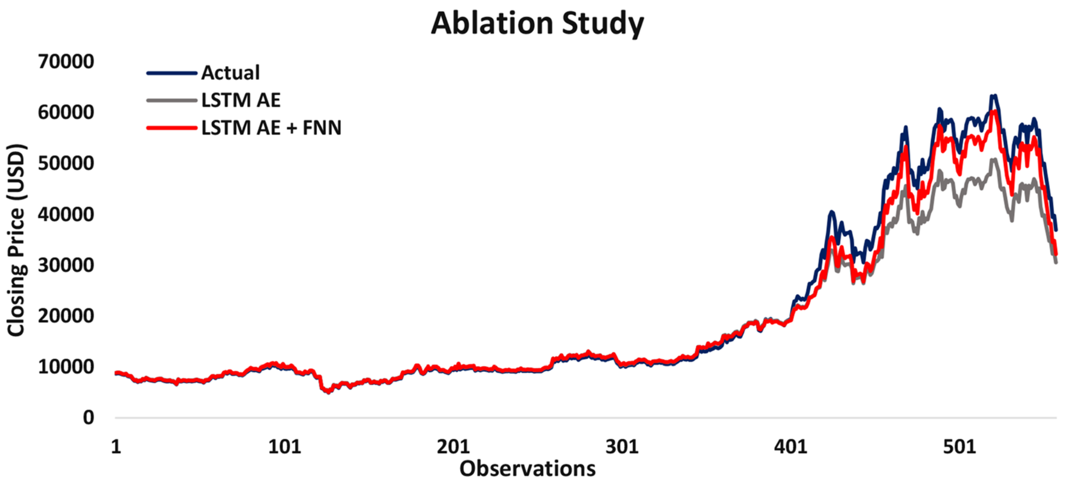

Fig. 5 Comparison of predicted values with and without the FNN loss component on testing set (Grey line represents a simple LSTM Autoencoder with MSE loss, Red line represents LSTM Autoencoder with FNN loss)

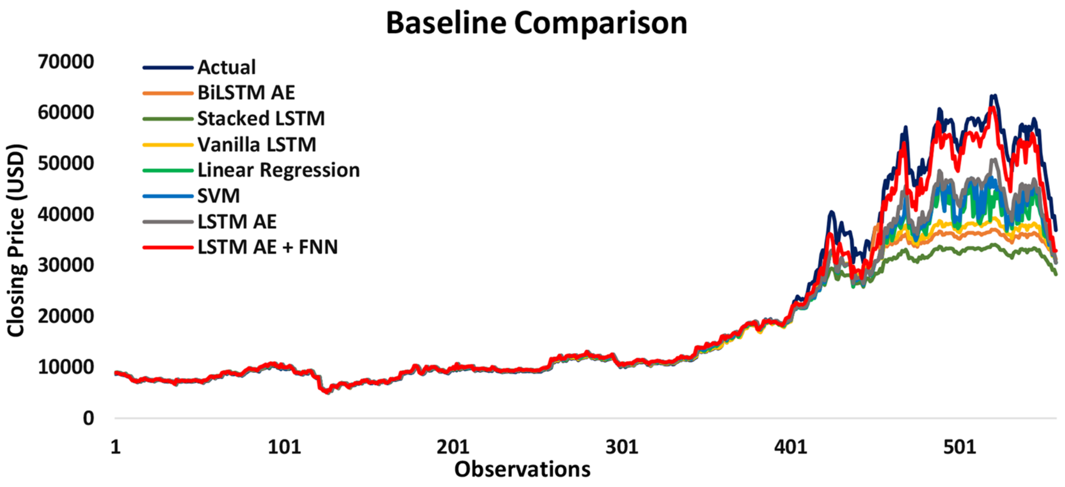

Fig. 6 Baseline Comparison on the testing set

FNN regularizer overcomes and leads to convergence with the actual prices.

We compare our method with existing baseline methods for time-series prediction. We re-implement the existing works on our dataset to compare the performances of state-of-the-art models. Widely used machine learning algorithms for cryptocurrency price prediction include Support Vector Machine (SVM), Logistic Regression (LR), and Neural Networks. We use these methods for prediction power comparison as these are well-established traditional forecasting methods. The comparison is made with four popular prediction methods, namely, Vanilla LSTM (Shahi et al. 2020), Stacked LSTM (Zanc et al. 2019), SVM (Chen et al. 2020), and LR (Chen et al. 2020). The parameters of these models are the same as described in respective works, except for the loss function. All other methods use the MSE loss function. The comparison in Fig. 6 highlights the accuracy and high convergence by the proposed architecture, which all other methods with the same configuration cannot achieve. Table 3 shows the comparison of RMSE values for all methods. The proposed method reaches the lowest error rates of 0.0120 and 0.5782 for training and testing phases. The model is superior to traditional machine learning algorithms because LSTM allows the deep learning model to automatically engineer real-world deterministic factors and utilize them for prediction, which is not the case for traditional

Table 3 Baseline comparison

Model	Training RMSE	Testing RMSE
Vanilla LSTM (Shahi et al. 2020)	0.0185	2.0106
Stacked LSTM (Zanc et al. 2019)	0.0373	2.5313
SVM (Chen et al. 2020)	0.0311	1.5100
LR (Chen et al. 2020)	0.0465	2.0123
LSTM AE (Ablation Study)	0.0440	1.3056
LSTM AE + FNN	**0.0120**	**0.5782**

Table 4 Test for robustness

	Vanilla LSTM	Stacked LSTM	Bi-LSTM AE	SVM	LR	LSTM AE	LSTM AE + FNN
Actual	12.13	11.94	11.83	12.06	12.17	12.01	8.08

methods. This accounts for the higher performance of the proposed method, overcoming the limitation of machine learning algorithms where such feature engineering becomes difficult due to the lack of deterministic datasets.

3.2 Test for robustness

To test for the robustness of the proposed method, we perform tests for robustness using statistical pairwise t-tests and evaluate the prediction performance using two test parameters: p-value and t-statistic value. For each model, we compare the prices predicted with the actual bitcoin prices. For all the models, the p-value obtained is less than 0.05, indicating a significant difference in actual and predicted prices, and we reject the null hypothesis for each model. Table 4 shows t-statistic values obtained for each model. The t-statistic value is lowest for the proposed method compared to existing baseline methods, demonstrating that the proposed LSTM Autoencoder model with FNN regularizer is highly robust and accurate.

4 Conclusion

This research contributes significantly to the field of Bitcoin price prediction by employing a novel LSTM Autoencoder prediction model that uses FNN as a regularizer in the loss function. We identify two deterministic factors impacting Bitcoin price fluctuations: behavioral investment and blockchain wallet users. These factors have significantly influenced price variations but have not been identified or established in existing works. Building on the determinants listed by Liu et al. (2021), we address the research gap in the literature. Given the many factors involved, which are dispersed across a multi-dimensional space, we propose a novel LSTM method that includes feature engineering to automatically select real-world deterministic factors. The proposed method significantly outperforms traditional prediction algorithms, providing robust results. We observe a substantial reduction in prediction error due to the FNN component. This research advances the development of effective deep learning methods for time-series prediction of financial commodities like Bitcoin's price. However, the lack of substantial data limits the performance of deep learning models. Future prospects include developing better-performing methods to bring predicted prices closer to actual prices.

Author contributions All authors contributed equally to the study conception, design and manuscript writing. All authors read and approved the final manuscript.

Funding The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data availability The dataset used for analysis will be available on request.

Declarations

Conflict of interests We declare that there is no financial or non-financial interests that are directly or indirectly related to the work submitted for publication.

References

Abarbanel HD, Brown R, Sidorowich JJ, Tsimring LS (1993) The analysis of observed chaotic data in physical systems. Rev Mod Phys 65(4):1331
Ahmed S, Hassan SU, Aljohani NR, Nawaz R (2020) FLF-LSTM: a novel prediction system using Forex loss function. Appl Soft Comput 97:106780
Antulov-Fantulin N, Guo T, Lillo F (2021) Temporal mixture ensemble models for probabilistic forecasting of intraday cryptocurrency volume. Decisions Econ Finan 44(2):905–940
Cao L, Mees A, Judd K (1997) Dynamics from multivariate time series. Physica D 121(1–2):75–88
Cavalli S, Amoretti M (2021) CNN-based multivariate data analysis for bitcoin trend prediction. Appl Soft Comput 101:107065
Chen Z, Li C, Sun W (2020) Bitcoin price prediction using machine learning: an approach to sample dimension engineering. J Comput Appl Math 365:112395
Chen L, Pelger M, Zhu J (2024) Deep learning in asset pricing. Manage Sci 70(2):714–750
CoinDesk (2021) CoinDesk: Bitcoin, Ethereum, Crypto News and Price Data https://www.coindesk.com/ (accessed on 2-Nov-2021)
Cretarola A, Figà-Talamanca G, Patacca M (2020) Market attention and Bitcoin price modeling: theory, estimation and option pricing. Decisions Econ Finan 43(1):187–228
De Angelis P, De Marchis R, Marino M, Martire AL, Oliva I (2021) Betting on bitcoin: a profitable trading between directional and shielding strategies. Decisions Econ Finan, 1–21
Demir E, Gozgor G, Lau CKM, Vigne SA (2018) Does economic policy uncertainty predict the Bitcoin returns? An empirical investigation. Finance Res Lett 26:145–149
Ding G, Qin L (2020) Study on the prediction of stock price based on the associated network model of LSTM. Int J Mach Learn Cybernet 11(6):1307–1317
Dyhrberg AH (2016) Bitcoin, gold and the dollar–A GARCH volatility analysis. Finance Res Lett 16:85–92
Figa-Talamanca G, Patacca M (2019) Does market attention affect Bitcoin returns and volatility? Decisions Econ Finan 42(1):135–155
Fraser AM, Swinney HL (1986) Independent coordinates for strange attractors from mutual information. Phys Rev A 33(2):1134

Gandal N, Hamrick JT, Moore T, Vasek M (2021) The rise and fall of cryptocurrency coins and tokens. Decisions Econ Finan 44(2):981–1014

Ghorbel A, Jeribi A (2021) Investigating the relationship between volatilities of cryptocurrencies and other financial assets. Decisions Econ Finan 44(2):817–843

Gilpin W (2020) Deep reconstruction of strange attractors from time series. Adv Neural Inf Process Syst 33:204–216

Grassberger P, Procaccia I (1983) Measuring the strangeness of strange attractors. Physica D: nonlinear phenomena 9(1-2):189–208.

Kantz H, Schreiber T (2004) Nonlinear Time Series Analysis. Cambridge University Press

Katsiampa P (2017) Volatility estimation for Bitcoin: a comparison of GARCH models. Econom Lett 158:36

Kennel MB, Brown R, Abarbanel HD (1992) Determining embedding dimension for phase-space reconstruction using a geometrical construction. Phys Rev A 45(6):3403

Liu M, Li G, Li J, Zhu X, Yao Y (2021) Forecasting the price of Bitcoin using deep learning. Finance Res Lett 40:101755

Lux T (2013) Inference for systems of stochastic differential equations from discretely sampled data: a numerical maximum likelihood approach. Ann Finance 9(2):217–248

Madan DB (2015) Asset pricing theory for two price economies. Ann Finance 11(1):1–35

Maknickienė N, Maknickas A (2012), May Application of neural network for forecasting of exchange rates and forex trading. In The 7th international scientific conference Business and Management (pp. 10–11)

Mari C, Mari E (2021) Gaussian clustering and jump-diffusion models of electricity prices: a deep learning analysis. Decisions Econ Finan 44(2):1039–1062

McNally S, Roche J, Caton S (2018), March 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

Na H, Kim S (2021) Predicting stock prices based on informed traders' activities using deep neural networks. Econ Lett 204:109917

Nakamoto S, Bitcoin A (2008) A peer-to-peer electronic cash system. Bitcoin.–URL: https://bitcoin.org/bitcoin. pdf, 4(2), 15

Nelson DM, Pereira AC, De Oliveira RA (2017), May Stock market's price movement prediction with LSTM neural networks. In 2017 International joint conference on neural networks (IJCNN) (pp. 1419–1426). IEEE

Phillips RC, Gorse D (2018), June Mutual-excitation of cryptocurrency market returns and social media topics. In Proceedings of the 4th international conference on frontiers of educational technologies (pp. 80–86)

Rezaei H, Faaljou H, Mansourfar G (2021) Stock price prediction using deep learning and frequency decomposition. Expert Syst Appl 169:114332

Sauer T, Yorke JA, Casdagli M (1991) Embedology J Stat Phys 65(3–4):579–616

Segnon M, Bekiros S (2020) Forecasting volatility in bitcoin market. Ann Finance 16:435–462

Shahi TB, Shrestha A, Neupane A, Guo W (2020) Stock Price forecasting with deep learning: a comparative study. Mathematics 8(9):1441

Sugihara G, May RM (1990) Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series. Nature 344(6268):734–741

Zanc R, Cioara T, Anghel I (2019), September Forecasting Financial Markets using Deep Learning. In 2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP) (pp. 459–466). IEEE