Can machine learning models better volatility forecasting? A combined method

***Abstract*—We apply and investigate the applications of classic GARCH family models, including GARCH, GJR-GARCH, TGARCH, and EGARCH, in the Bitcoin market. These traditional volatility forecasting models are effective in capturing general risk changes, however, cannot accurately cope with extreme market shocks. To solve this issue and enhance the volatility forecasting results, we use the Long short-term memory (LSTM) neural network to combine predictions from those GARCH-type models and historical volatility observations. We demonstrate that this deep learning method is useful for building an accurate volatility forecasting model for the Bitcoin market. More importantly, the LSTM can optimize information intake through the short- and long-memory states. Hence, compared with GARCH models that only consider predefined, limited timesteps, the new LSTM neural network model is more robust in adapting to market shocks and regime changes.**

***Index Terms*—Bitcoin, Volatility, Forecasting, LSTM, GARCH**

I. Introduction

Cryptocurrencies, despite many ongoing debates and controversies, continue to evolve. For instance, in 2022, there were over 22,000 different 'coins' existing (including the inactive and discontinued ones)[1]. Until December 2023, there are more than 9,000 active cryptocurrencies[2]. Some market intelligence predicts the overall capitalisation could reach USD5.03 trillion by 2028 that implies an approximately compound annual growth rate of 30.4% from now on. In light of all these, Bitcoin continues to occupy the top spot with a whopping market cap of around USD561.3bn currently[3].

The Bitcoin market is highly speculative. Many professionals believe that, until today, Bitcoins, as well as a wider range of cryptocurrencies, are not isolated from the traditional banking system. For instance, the crypto-focused banks, including Silvergate Bank, Silicon Valley Bank and Signature Bank, were shut down following the bankruptcy of crypto exchange FTX. These failures are proven to be associated with the banks' crypto holdings and bring significant distortion to the financial market stability. Inevitably, to better predict Bitcoin volatility is in everyone's interest and beneficial for the economy.

The topic of volatility forecasting has accumulated rich literature since [1] proposes the idea of conditional heteroskedasticity and forms the general ARCH model. [2] extends it to have half autoregressive conditional heteroskedasticity that becomes the original GARCH model. Together with many variations, the family of GARCH models have become the most widely used model to handle three hard properties of financial time series: asymmetric extreme values, fat-tails, and volatility clustering (see [1], [2], [3], [4], [5]). There are also efforts to expand the GARCH models to forecast realized volatility. For instance, [6] proposes a realised GARCH model to animate the ARMA model to obtain the conditional variance and an intraday realised volatility measure. They provide an empirical example of Down Jones Industrial Average stocks and an exchange traded index fund to demonstrate an improvement over the previous GARCH models that only consider information on a daily frequency. Later, [7] introduces the realized EGARCH model that allows for eight different realized volatility measures and finds that the realized measures improve the log-likelihood performance of both in-sample and out-of-sample returns. But not all realized measures perform equally well. They argue that, for the out-of-sample, the best model performance is achieved through a combination of two realized measures: the daily range realized variance and the realized variance based on two-minute sampling. This is primarily due to the microstructure noise interfering with the model capturing the conditional volatility of returns. Similar efforts for realized volatility forecasting can be seen in [8].

The Bitcoin market, compared with the stock market, is much more accessible for investors due to the benefits of the decentralized blockchain technology such as low entry barrier, transparency, effective transaction cost etc. Meanwhile, the market is proved to be volatile. Several researchers opt to estimate its volatility and forecasting using GARCH family models. For example, [9], [10], [11] have tried more than 200 different GARCH models, and [12] compared the performance of GARCH models with heterogeneous autoregressive (HAR) models —both suggest that GARCH models are useful and efficient in forecasting the volatility of the Bitcoin market.

However, Bitcoin returns exhibit prominent stochastic volatility with unique tail features (see Figures 1 and 2). On one hand, the return distribution is not normal (i.i.d) given the high peak despite the visible symmetry. On the other hand, it resembles hyperbolic tails in Figure 2b. While differing from the common fat-tails, it has rather long and thin tails on both left and right sides, which result in a rather large kurtosis. These features suggest that there are extreme values in the time series and this can be rightly observed in the the Bitcoin price and implied volatility in Figure 1. These features expose potential volatility prediction errors with GARCH models because their good performance is

[1]https://commonslibrary.parliament.uk/research-briefings/cbp-8780/

[2]https://www.statista.com/statistics/863917/number-crypto-coins-tokens/

[3]See https://fintechmagazine.com/articles/top-10-cryptocurrencies-in-2023-by-market-capitalisation.

theoretically conditioned only if the time series do not contain extreme values as suggested in [13]. Nevertheless, GARCH modelling is valuable, as a fundamental method, for volatility estimation and forecasting. Therefore, in this paper, we aim to improve GARCH modelling by addressing the specific issues that appear in the Bitcoin market.

We believe that the GARCH-type models well-fitted with the Bitcoin returns could, to some extent, provide trustworthy implied volatility estimations and predictions. In addition, the fast developing deep learning tools also suggest ideas of applying neural networks or other learning models for financial market forecasting. One candidate is the Long Short-Term Memory (LSTM) model that stands out as a superior model for sequential data [14]. A series of studies suggest that LSTM could improve a specific time series model (e.g., AR model, GARCH model) performance, especially their forecasting ability when extreme values present. LSTM also has capacity to incorporate multiple market provisions, such as trading volumes [15], [16], thus forms a more powerful modeling tool. So, we suggest to further take the implied volatility given by the well-established GARCH models as input to the LSTM to generate an enhanced volatility forecasting solution. The advantage of LSTM in adapting various memory-length and decaying for different features also mirrors our thoughts of mimicking the distinctive clustering features and responses of extreme values together with multiple GARCH models. Regarding Bitcoin volatility forecasting, most previous literature train models using realised volatility as the target [9], [17], [12], [18], [19]. While [20] argues that implied volatility is a better indicator than realised volatility after sorting out the overlapping and high autocorrelation issues in the data sampling procedure (also see [21]). Hence, given the new reliable Bitcoin implied volatility index BitVol, produced by T3 Index[4], we train our model targeting on forecasting implied volatility.

We establish two LSTM neural network models for 1-day and 5-day implied volatility forecasting, respectively. These models outperform the GARCH family models in both training and testing periods. In particular, evidenced by the testing dataset, we successfully reduce the prediction percentage error from over 10% (across all GARCH-type models) to 5.70% for 1-day forecasting; and from over 10% (across all GARCH-type models) to 8.22% for 5-day forecasting. Overall, the Bitcoin volatility forecasting literature is not vast and, to our best knowledge, there were limited studies considering addressing the model predictability through both achieving model fit and automation using a deep learning machine, in other words, a combined method. The contributions are in multiple folds. First, it accounts for the computational efficiency because both GARCH models and the LSTM are quick to compute and run. Second, the forecasting accuracy is improved because of the good 'preliminary' forecasting results generated by well-fitted GARCH models, coupled with the proficiency of LSTM in handling sequential data. Third, the new model is easy to use because analysts and traders are familiar with the GARCH models and their properties, the single-layer LSTM neural network is linear-like, and more importantly, the LSTM is accurate and robust through the testing period.

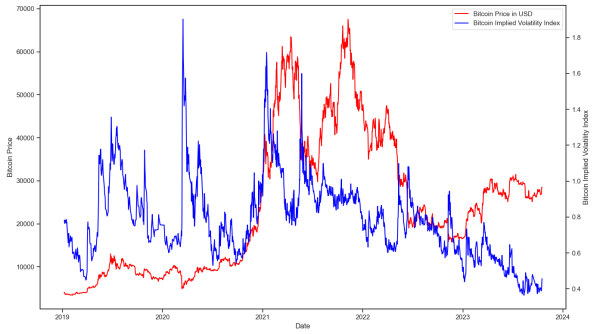
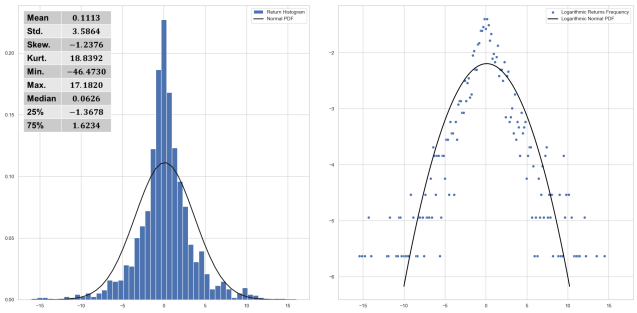


Fig. 1: Bitcoin prices and implied volatility index over time



(a) Normal dist. PDF (b) Logarithmic Normal PDF

Fig. 2: Statistics properties of Bitcoin return

The rest of this paper is organized as follows: Section II provides an overview of the volatility research for Bitcoin. Sections III and IV presents the data and methods used. Section V shows the results, section VI discusses the model interpretations, and section VII draws the conclusion and discusses future research.

II. Literature Review

From the risk management and trading perspective, Volatility forecasting is, arguably, one of the most important problems. It has been studied with a long history since [21], [22]. Technically, scholars suggest volatility is more predictable than daily returns (see [23], [24]).

GARCH-type modelling is probably the most commonly used method[5] to estimate and forecast volatility due to its advantage of coping with the time series features such as the persistence of volatility, the mean-reversion behaviour, the asymmetric impact of negative vs. positive return innovations. Since [1] proposed the ARCH model in 1982 and [2] established the original GARCH model, a whole range of estimation

[4]https://t3index.com/indexes/bit-vol/

[5]Alternatively, stochastic volatility (SV) models like the one seen in [25] can be used.

models have emerged including the GJR-GARCH [4], threshold GARCH [3], Exponential GARCH [5] and so on, aiming to capture volatility clustering, heteroskedasticity, asymmetric shock and leverage effect.

Bitcoin volatility estimation and forecasting are exception in attracting interest and many naturally opt for GARCH modelling. For example, [9] apply 11 types of GARCH models to estimate Bitcoin's volatility using a sample from 2010 to 2016 and find that the AR-CGARCH model fits the best. [17] fit more than 1000 GARCH models to four different cryptocurrency data (including Bitcoin) to obtain a one-step ahead prediction of Value-at-Risk and Expected-Shortfall. In general, simple GARCH models do not perform well. Instead, if some improvement is made to account for asymmetry or regime switch (e.g., Marko-Switching), such a modified GARCH model would fit much better. [12] compare the several GARCH and two heterogeneous autoregressive (HAR) models to predict Bitcoin volatility. HAR models based on realized variance computed from high frequency data show forecasting superiority to GARCH-based implied volatility prediction using daily data, especially for the short-term forecasting. [26] compare forecast performance of GARCH, HAR and ARIMA models against the implied volatility derived from option pricing. They take the thought that implied volatility is more meaningful than the realised volatility as it focuses more on future [20]. They conclude that the implied volatility is generally a good measure for market volatility in the near future (e.g., 7-, 10- and 15-day) while HAR has the best 1-day forecasting.

Lately, new research starts to incorporate machine learning and deep learning methods into the traditional volatility models. For instance, [11] explore 110 different GARCH-type models, finding varied out-of-sample forecasting performance, and proposes a combined method consisting of GARCH and the Support Vector Machine. The trained best combination of forecasting models significantly improves the Bitcoin volatility prediction.

Further, deep learning models prevail because of their ability to handle mass data and non-linear relationships among variables, but they need good inputs to ensure good quality of training. As [27] pointed out when the input was not accurate, an advanced deep learning model will not necessarily give you better forecasting performance. Specifically, they input the realized volatility computed from the daily squared returns and historical Garman-Klass volatility into the Recurrent Neural Network (RNN); and it turns out that both RNNs perform worse than a simple GARCH(1,1) forecasting. But researchers continue to prefer the combined methods for its computational efficiency, potential for automation and other benefits as mentioned before. Another key work can be identified in [16] which mixes various GARCH models with a LSTM model, though they are examining some highly liquid stocks' volatility, the LSTM forecasting accuracy is evidently improved. [15] try a similar idea on combining LSTM and AR models and find better predictions of the Bitcoin price movements. Similarly, [18] confirm that their combined LSTM-GARCH model can better forecast Bitcoin volatility, through their forecasting target is the realized volatility. When [19] consider other factors such as the US daily news index, they conclude that the similar combined method outperforms traditional forecasting methods.

To sum up, training forecasting models through deep learning is currently favoured but it requires reliable input features. Various volatility estimations could serve as input, while accuracy is the key issue. These prove that our idea of improving the existing GARCH estimation through a low-cost but effective tuning of symmetric/asymmetric model selection and distributional assumption would be highly feasible to generate better inputs. Also, it is worth noting again that GARCH models and their volatility estimations are more computationally efficient than many other techniques that involve complex terms, such as realized volatility computations, in the stochastic volatility process. We then operate a single-layer LSTM that could efficiently provide enhanced volatility forecasting.

III. Methodology

We explore the implementation and performance of implied volatility forecasting using GARCH family models. To enhance the forecasting results, we build a Long Short-Term Memory (LSTM) neural network model in which the GARCH family models' predictions and historical implied volatility data are set as features.

A. GARCH family models

The GARCH family models are constructed to introduce mathematical techniques that deal with stochastic volatility:

$$\varepsilon_t = \sigma_t Z_t, \tag{1}$$

where ε_t is the return residuals, σ_t is the stochastic volatility term, Z_t is the noise term of mean zero, variance one. The innovation distribution of Z_t is usually selected from a normal distribution or a Student-T distribution.

The standard GARCH model is defined by [2]. See below for GARCH(1,1):

$$\begin{aligned} \sigma_t^2 &= \omega + \alpha\varepsilon_{t-1}^2 + \beta\sigma_{t-1}^2, \\ &\omega > 0,\ \alpha > 0,\ \beta > 0. \end{aligned} \tag{2}$$

[4] argues that volatility responds to positive and negative residuals at varying levels. Hence, they propose the GJR-GARCH model by adding a term of negative residuals. We use the GJR-GARCH(1,1) version in this paper:

$$\begin{aligned} \sigma_t^2 = \omega + \alpha\varepsilon_{t-1}^2 + \gamma\varepsilon_{t-1}^2\mathbb{1}_{\{\varepsilon_{t-1}<0\}} + \beta\sigma_{t-1}^2, \\ \omega > 0,\ \alpha > 0,\ \beta > 0,\ \gamma \in \mathbb{R}, \end{aligned} \tag{3}$$

where $\mathbb{1}_A$ is an indicator function such that $\mathbb{1}_A = 1$ if A is satisfied, otherwise $\mathbb{1}_A = 0$. The GJR-GARCH model is reduced to the GARCH model if γ is zero.

To model the asymmetric volatility's responses, [3] introduces an alternative way to model the asymmetric responses of positive and negative return shocks, which is the Threshold GARCH (TGARCH) model. In this model, volatility responds

to absolute residuals rather than squared residuals, see Equation (4) for TGARCH(1,1).

$$\sigma_t^2 = \omega + \alpha|\varepsilon_{t-1}| + \gamma|\varepsilon_{t-1}|\mathbb{1}_{\{\varepsilon_{t-1}<0\}} + \beta\sigma_{t-1}^2, \quad \omega > 0,\ \alpha > 0,\ \beta > 0,\ \gamma \in \mathbb{R}. \tag{4}$$

The Exponential GARCH (EGARCH) proposed by [5] models the logarithmic variance. This model assures that the variance is non-negative. In this paper, we use the symmetric version of EGARCH(1,1):

$$\ln \sigma_t^2 = \omega + \alpha(|Z_{t-1}| - \mathbb{E}|Z_{t-1}|) + \beta \ln \sigma_{t-1}^2, \quad \omega > 0,\ \alpha > 0,\ \beta > 0. \tag{5}$$

B. Long short-term memory (LSTM) neural network model

While GARCH family models provide valuable volatility forecasts, there still exist gaps and flaws that require addressing. Observing the variations in the models above, we conclude that there is no universal form that suits every situation. This is the first challenge we encounter when applying those classic models to a new market like Bitcoin. Moreover, even though we select a model using mathematical techniques, the return time series properties may change over time as the market continues to evolve. To address these issues and build an accurate and sustainable volatility forecasting model, we use a neural network, called LSTM, to combine outcomes from different GARCH models.

The LSTM model is a kind of recurrent neural network (RNN), particularly designed for sequential data inputs. RNNs are distinguished by "memory" – they take information from prior inputs to current nodes. Hence, it is good at dealing with temporal problems. LSTM is enhanced from the basic RNNs by involving a forget gate, which drops 'out dated' information from prior nodes (see Figure 3).

In the LSTM, an input $\boldsymbol{X}_{T\times N}$ is constructed by T look-back timesteps and N features. On each timestep, the LSTM cell generates a hidden state $(H_t)_{1\times m}$, where m is the number of units in the LSTM layer. By processing X_t from the earliest to the latest timesteps, information flows through the cell state $\{(C_t)_{1\times m} : t = 1, 2, ..., T\}$. From prior time $t-1$ to t, the current cell state takes information from the previous cell state and new input:

$$C_t = f_t \otimes C_{t-1} + i_t \otimes N_t,$$

where $\otimes$ denotes the Hadamard product, $f_t \in [0,1]^m$ is the forget gate such that $f_t = 0$ means removing everything in C_{t-1}, N_t is new information given by new input and memory carried from previous hidden state, $i_t \in [0,1]^m$ is the input gate such that $i_t = 1$ means taking all information in N_t. The new information carries the previous hidden state H_{t-1} and the new input X_t:

$$N_t = \mathcal{L}(X_t \cdot u_c + H_{t-1} \cdot w_c + b_c)$$

where $(u_c)_{N\times m}$ and $(w_c)_{m\times m}$ are weights of the current input and the previous hidden state, respectively, $(b_c)_{1\times m}$ is a bias parameter. The activation function $\mathcal{L}(x)$ introduces a scaling of the addition and subtraction of information. Now, we focus on the most important part in the LSTM, the output H_t. It is given by an output gate $o_t \in [0,1]^m$ and the cell state C_t:

$$H_t = o_t \otimes \mathcal{L}(C_t).$$

If $o_t = 1$, the LSTM cell sends out all information in the cell state as a prediction; otherwise 'discounted' information is used as a forecast result. Note that the same activation function should be used for the new information N_t and the output H_t.

The forget, input and output gates all consider both new input X_t and the prior output H_{t-1}:

$$f_t = \Phi(X_t \cdot u_f + H_{t-1} \cdot w_f + b_f),$$
$$i_t = \Phi(X_t \cdot u_i + H_{t-1} \cdot w_i + b_i),$$
$$o_t = \Phi(X_t \cdot u_o + H_{t-1} \cdot w_o + b_o),$$

where $(u_f)_{N\times m}$, $(w_f)_{m\times m}$, $(u_i)_{N\times m}$, $(w_i)_{m\times m}$, $(u_o)_{N\times m}$ and $(w_o)_{m\times m}$ are weights; $(b_f)_{1\times m}$, $(b_i)_{1\times m}$, $(b_o)_{1\times m}$ are bias parameters; and $\Phi(\cdot)$ is the Sigmoid function below:

$$\Phi(x) = \frac{1}{1+\exp(-x)}.$$

The advantage of LSTM lies in its ability to learn how much old information should be stored in the long memory state C_t, and what can be ignored. Hence, the hidden state H_t indicates short memory. The structure of the LSTM cell is shown in Figure 3.

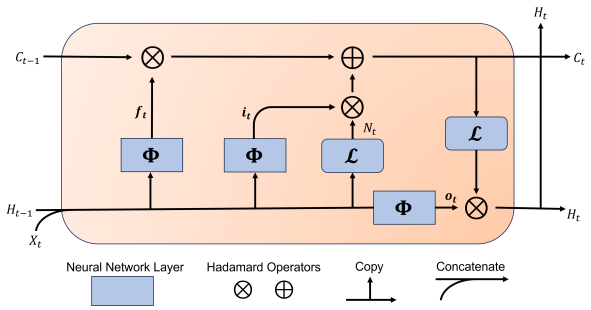


Fig. 3: LSTM Neural Network Cell

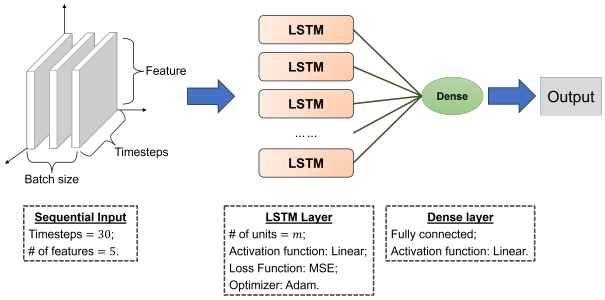


Fig. 4: LSTM Neural Network Architecture

We use a simple LSTM neural network architecture. The sequential input X_t is constructed by a 30-day look-back period of five features: 1-day forecast given by GARCH(1,1), GJR-GARCH(1,1), TGARCH(1,1) and EGARCH(1,1), and

past Bitcoin implied volatility (BitVol) values from $t-30$ to $t-1$. The target Y_t is the BitVol at t. Other technical specifics are summarized in Figure 4. Here, we would like to highlight that we use a linear activation function, which is uncommon in LSTM model. The rationale behind this is that all the input features are on the same scale as the target and should not deviate far away from the target. Hence, we believe the training will be more efficient without scaling or shape changes.

C. Performance metrics

To examine the forecasting results of models introduced in previous sections, we use the performance metrics in Equations 6 and 7, respectively. Recall that we use MSE as the loss function, which is the squared RMSE. Hence, RMSE is theoretically minimized when training the model.

$$RMSPE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{Y}_i}{\hat{Y}_i}\right)^2} \tag{6}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2} \tag{7}$$

where $\{Y_i : i = 1, 2, ..., n\}$ is the forecast values and $\{\hat{Y}_i : i = 1, 2, ..., n\}$ is the target values.

RMSPE would be more effective than other measures in capturing the degree of errors in relation to the desired target values. Additionally, RMSPE would penalize significant errors more severely than standard RMPE , which this paper needs. So, RMSPE has priority in this paper when comparing the performance of model prediction.

IV. Data

Bitcoin daily prices are obtained from Yahoo Finance. We choose the BitVol index as the outcome for our targeted volatility predictions. BitVol, produced by T3 Index, is a daily updated index tracking expected 30-day implied volatility in Bitcoin. Our data is from 7 January 2019 to 17 October 2023, in total 1745 trading days. Note that the Bitcoin market opens 24/7 and does not close on weekends or public holidays. The Bitcoin daily close price in U.S. dollars is recorded in the GMT zone.

The BitVol is a new index, so, has some missing values in weekends and holidays before June 2020. Two different treatments are applied when it is used in the training dataset and as the target. We fill in missing data by linear interpolation for the target. Apparently, this method is not suitable as the next value may not be observable. Hence, we fill in previous available values when using BitVol as a training feature. We split the data into training and test data using an 80-20 ratio. The training period is from 7 January 2019 to 1 November 2022, in total 1395 observations. The testing period is from 2 November 2022 to 16 October 2023, in total 349 observations. Note that the training period for the LSTM is 30 days shorter as we take sequential inputs with 30 timesteps.

V. Results

We first determine the mean process for return residuals ε_t. We examine the ACF and PACF in Figures 5, so, confirm that no autocorrelations in returns and return residuals. Hence, we find return residuals through demeaning: $\varepsilon_t = x_t - \mu$, where μ is the average of returns in the training dataset.

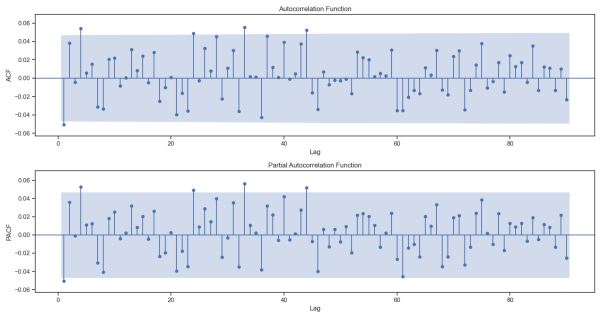


Fig. 5: ACF and PACF of returns.

	ω	α	β	γ	ν
GARCH	0.236	0.070	0.930		2.0961
GJR-GARCH	0.191	0.077	0.937	−0.028	2.994
TGARCH	0.095	0.079	0.927	−0.012	3.154
EGARCH	0.066	0.166	0.988		2.805

TABLE I: Calibration results of GARCH family models

	Training dataset		**Testing dataset**	
	RMSPE	RMSE	RMSPE	RMSE
GARCH	16.22%	15.86	12.71%	6.96
GJR-GARCH	16.01%	14.88	10.90%	6.022
TGARCH	16.85%	15.26	10.51%	5.83
EGARCH	14.30%	12.61	12.49%	7.28
LSTM	5.13%	5.08	5.70%	3.13

TABLE II: 1-day implied volatility forecast performance

The calibration results of GARCH family models, as well as the degree-of-freedom of the Student-T innovation distribution of each model, are presented in Table I. We use these models to do 1-day volatility forecast and 5-day average volatility forecast (i.e., $\frac{\sum_{i=1}^{5}\sigma_{t+i}}{5}$). The 1-day ahead volatility σ_{t+1} is computed by ε_t and σ_t which is known at t. However, volatility from 2-day ahead will need unobservable noise terms. We use two techniques to address this issue: one is the simulation technique that draws Z_t from corresponding innovation distributions; the other one is the bootstrap technique that draws noise terms from the random sampling given by the training dataset. Although these techniques can give further volatility forecasting, we do not expect either of them to produce a reliable result in the long-run. Hence, we limit the experiment to a short term of 5 days. It is worth noting that bootstrap results are not available for the training dataset due to the lack of random samples.

	Training dataset		Testing dataset			
			RMSPE		RMSE	
	RMSPE	RMSE	Sim.	Boot.	Sim.	Boot.
GARCH	15.79%	15.67	14.76%	14.35%	8.09	7.91
	(8.63×10^{-3})	(6.83×10^{-1})	(5.73×10^{-3})	(3.62×10^{-4})	(3.35×10^{-1})	(2.30×10^{-2})
GJR-GARCH	15.65%	14.68	12.51%	12.24%	6.99	6.88
	(7.14×10^{-3})	(5.15×10^{-1})	(8.61×10^{-3})	(2.86×10^{-4})	(4.70×10^{-1})	(1.70×10^{-2})
TGARCH	16.49%	15.18	11.22%	11.17%	6.19	6.18
	(1.46×10^{-4})	(9.12×10^{-3})	(1.35×10^{-3})	(1.24×10^{-4})	(7.16×10^{-2})	(6.75×10^{-3})
EGARCH	188791%	1.42×10^{5}	1518.46%	13.27%	8.53×10^{2}	7.97
	(1.26×10^{108})	(1.03×10^{110})	(1.83×10^{28})	(2.04×10^{-4})	(8.45×10^{29})	(1.32×10^{-2})
LSTM	6.46%	6.22	8.22%		4.47	

TABLE III: 5-day average implied volatility forecast performance

The performance metrics are in Tables II and III. For the 5-day forecasting, we carry out 1000 runs to examine the robustness of the methods. The values shown in Table III are the median and standard deviation of performance metrics, with the latter in parentheses. We do not find clear differences on 1-day forecasting using these GARCH-type models. EGARCH performs the best for the training dataset, but does not stand out in the testing stage. We observe an unusual result that all models exhibit much better performance in testing, particularly with regard to the RMSE metric. In Figure 6, we examine the error $Y_i - \hat{Y}_i$ for each target $\hat{Y}_i$ and confirm that the 'better' performance is a result of coincidentally avoiding high volatility levels during the testing period. All four models 'fail' when volatility exceeds $120(\%)$, with GARCH, GJR-GARCH and EGARCH either over- or under-predict the volatility. TGARCH exhibits an even harmful systemic error, mainly producing under-predictions; moreover, as volatility increases, so does the bias. We believe the same explanation applies to the performance of 5-day forecasting, although we cannot observe it due to the unstable outcomes resulting from the simulation method. Another important observation is that, in the 5-day forecasting given by simulation methods, EGARCH explodes, reflected by the large standard deviations. This may be due to the log-linearity form, which produces extremely high volatility when exponentiating results given by some extreme tail innovations.

In comparison, we also investigate the errors given by the LSTM model (see Figure 7). Obviously, the deep learning technique effectively eliminates systemic errors introduced in the inputs and ensures a more accurate prediction close to the target values. Concerning the over-fitting issue commonly observed in deep learning models, we confirmed that the performance on the testing dataset closely aligns with that of the training dataset. In Figures 8 and 9, we find that, overall, the forecasting given by the LSTM models is accurate. While it is challenging to train a 5-day forecasting model as effectively as the 1-day model. Lacking updated information, the model may fail to catch up with the market fluctuation even in short-term. For example, we observe some unstable predictions in 2021; and the model begins to show signs of deviating from the target shortly within a year of testing.

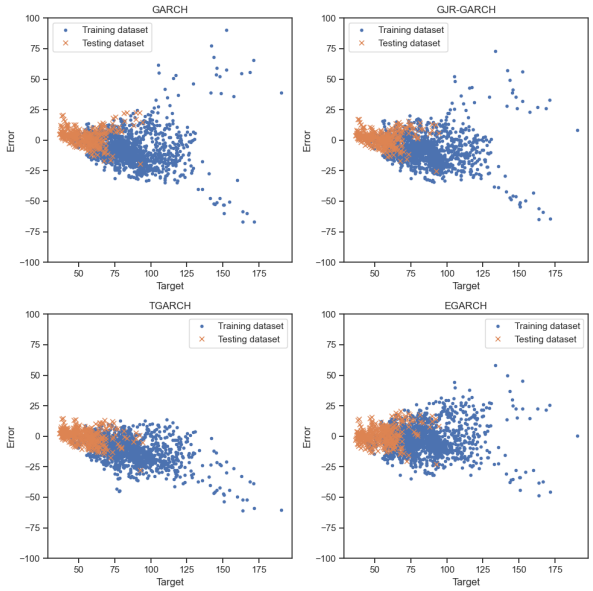


Fig. 6: GARCH family 1-day errors vs. Implied volatility

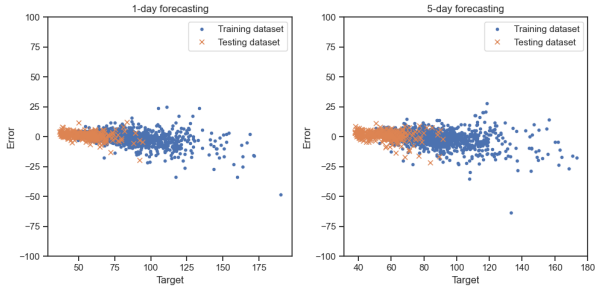


Fig. 7: LSTM errors vs. Implied volatility

VI. Discussion

In studies on deep learning applications, we notice that there is not much focus on interpreting how models work. Due to the model complexity, it is challenging to extract the impact of a feature or clearly draw the memory decaying patterns for LSTM. However, we still think it is worthwhile to exhibit deeper insights of our model. Firstly, we look into the weights

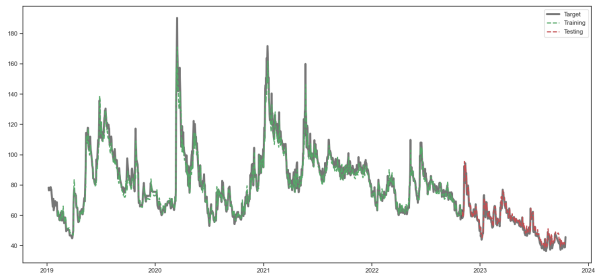


Fig. 8: LSTM 1-day implied volatility forecasting

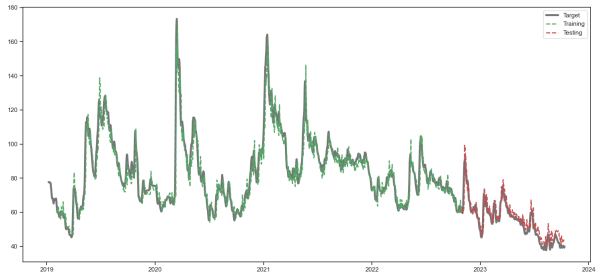
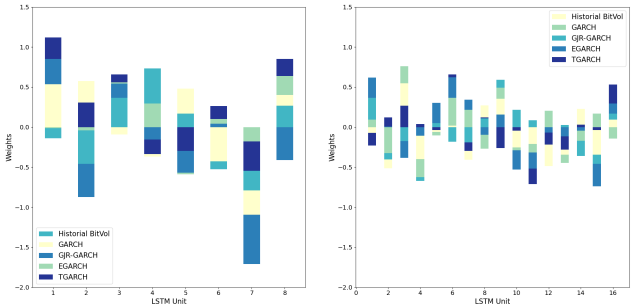


Fig. 9: LSTM 5-day implied volatility forecasting

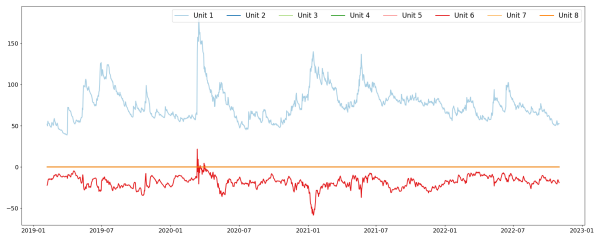
u_c that carry input X_t to new information in each LSTM unit (see Figure 10). We see that each LSTM unit handles a specific linear combination of the inputs. In Figure 10a, in particular, the 1st and 7th units mainly focus on positive and negative weighted averages, respectively. We observe that the 1-day forecasting model applies 'strong' negative weights to the GJR-GARCH input (e.g., the 2nd, 7th and 8th units), whereas it is not the case for the 5-day forecasting model. Another noteworthy finding is that the weights in the 5-day forecasting model are much smaller than those in the 1-day forecasting model, indicating that the former 'tunes' the inputs more conservatively.



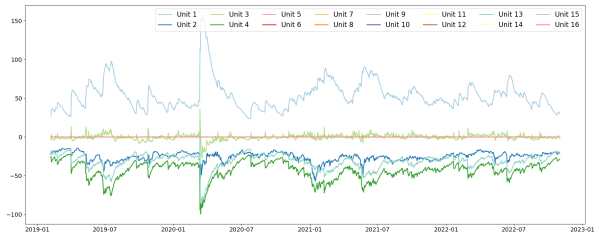
(a) 1-day forecasting model (b) 5-day forecasting model

Fig. 10: LSTM new information weights on inputs

Knowing how the new inputs are passed into the forecasting is just the beginning of the story. More importantly, after transferring the X_t to new information N_t, it will pass through different gates in sequential timesteps (in total 30 timesteps in our model) until reaching the final output. As the inputs and memory tangle together when 'forming' the gates and longer memories, we cannot elaborate on all cases. However, the LSTM layer outputs do shed lights on the roles of the units. In Figure 11, we observe that not all units have 'effective' outputs to the dense layer, while we will not conclude that those units are useless as they may control the gates.



(a) 1-day forecasting model



(b) 5-day forecasting model

Fig. 11: LSTM layer outputs of training dataset

According to this observation, we run a rough analysis about the feature impact by multiplying u_c —excluding the columns of units that only give outputs equal or close to 0, by the dense layer weights of the same units. We obtained the overall feature weights in Table IV. We conclude that the forecasting results are mainly driven by the historical BitVol index. Minor adjustments are given by different GARCH family models, with GARCH contributing the most. In the 1-day forecasting model, GJR-GARCH and TGARCH are used as a kind of 'reversal' indicator. EGARCH shows almost no impact in the 5-day forecasting model, indicating that it is not suitable for long-run predictions.

VII. Conclusion

In this paper, we combine the deep learning technique with classic GARCH-type models to obtain a hybrid forecasting model that provides more accurate volatility forecasting in the Bitcoin market. Among many time series models, the GARCH family is widely used in volatility forecasting due to their stable performance and efficiency. While our application results for GARCH(1,1), GJR-GARCH(1,1), TGARCH(1,1) and EGARCH(1,1) demonstrate that they do not produce accurate forecasting results, even exhibit systemic errors. Hence, we involve the LSTM model to improve the forecasting precision. By training a model with the capability to selectively retain and discard memories, our LSTM neural network model reduces the 1-day and 5-day forecasting error from over 10% to 5.70% and 8.22%, respectively. This is not only has a significant improvement in overall forecasting performance,

	Hist. BitVol	GARCH	GJR-GARCH	TGARCH	EGARCH
1-day model	0.7133	0.2638	−0.0427	−0.0645	0.1642
5-day model	0.7231	0.1382	0.1621	0.1232	0.0073

TABLE IV: LSTM features impact

but also demonstrates an effective solution to the GARCH model's error in under-predicting extreme volatility.

We conclude three key findings from our experiment. Firstly, the Bitcoin market shows similar time series properties as the stock market, so, classic modelling techniques such as GARCH models are applicable. However, when using those models for volatility forecasting, the performance vary. Moreover, GARCH-type models do not effectively handle the more frequent high-volatility conditions in the Bitcoin market. Secondly, although the none of the GARCH-type models we applied is adaptive to extreme market shocks, they can serve as good and stable input features in deep learning models. Our model demonstrates superior performance compared to most previous literature studying similar forecasting issues, attributed to the effective input from GARCH models and historical BitVol. Lastly, the LSTM neural network possesses advantages in financial time series forecasting due to its inherent design for handling sequential data. As the market absorbs shocks and information from the past, a model that strategically builds long and short memories is particularly useful for forecasting. This explains why the LSTM can tune the inaccurate and old volatility from various sources to more accurate and robust predictions. A limitation of this study may lie on the short history of the Bitcoin market. With expanded observations in the future, or involving high-frequency data, the model may be improved by considering more diverse information such as market sentiment.

REFERENCES

[1] R. F. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation," *Econometrica: Journal of the econometric society*, pp. 987–1007, 1982.

[2] T. Bollerslev, "Generalized autoregressive conditional heteroskedasticity," *Journal of econometrics*, vol. 31, no. 3, pp. 307–327, 1986.

[3] J.-M. Zakoian, "Threshold heteroskedastic models," *Journal of Economic Dynamics and Control*, vol. 18, no. 5, pp. 931–955, 9 1994.

[4] L. R. GLOSTEN, R. JAGANNATHAN, and D. E. RUNKLE, "On the Relation between the Expected Value and the Volatility of the Nominal Excess Return on Stocks," *The Journal of Finance*, vol. 48, no. 5, pp. 1779–1801, 12 1993.

[5] D. B. Nelson, "Conditional Heteroskedasticity in Asset Returns: A New Approach," *Econometrica*, vol. 59, no. 2, p. 347, 3 1991.

[6] P. R. Hansen, Z. Huang, and H. H. Shek, "Realized GARCH: a joint model for returns and realized measures of volatility," *Journal of Applied Econometrics*, vol. 27, no. 6, pp. 877–906, 9 2012.

[7] P. R. Hansen and Z. Huang, "Exponential GARCH Modeling With Realized Measures of Volatility," *Journal of Business & Economic Statistics*, vol. 34, no. 2, pp. 269–287, 4 2016.

[8] T. G. Andersen, T. Bollerslev, and N. Meddahi, "Realized volatility forecasting and market microstructure noise," *Journal of Econometrics*, vol. 160, no. 1, pp. 220–234, 1 2011.

[9] P. Katsiampa, "Volatility estimation for Bitcoin: A comparison of GARCH models," *Economics letters*, vol. 158, pp. 3–6, 2017.

[10] A. H. Dyhrberg, "Bitcoin, gold and the dollar–A GARCH volatility analysis," *Finance Research Letters*, vol. 16, pp. 85–92, 2016.

[11] S. Aras, "On improving GARCH volatility forecasts for Bitcoin via a meta-learning approach," *Knowledge-Based Systems*, vol. 230, p. 107393, 2021.

[12] L. Ø. Bergsli, A. F. Lind, P. Molnár, and M. Polasik, "Forecasting volatility of Bitcoin," *Research in International Business and Finance*, vol. 59, p. 101540, 2022.

[13] P. H. Franses and D. Van Dijk, "Forecasting stock market volatility using (non-linear) Garch models," *Journal of Forecasting*, vol. 15, no. 3, pp. 229–235, 4 1996.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] C.-H. Wu, C.-C. Lu, Y.-F. Ma, and R.-S. Lu, "A new forecasting framework for bitcoin price with LSTM," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2018, pp. 168–175.

[16] H. Y. Kim and C. H. Won, "Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models," *Expert Systems with Applications*, vol. 103, pp. 25–37, 2018.

[17] G. M. Caporale and T. Zekokh, "Modelling volatility of cryptocurrencies using Markov-Switching GARCH models," *Research in International Business and Finance*, vol. 48, pp. 143–155, 4 2019.

[18] M. Zahid, F. Iqbal, and D. Koutmos, "Forecasting Bitcoin Volatility Using Hybrid GARCH Models with Machine Learning," *Risks*, vol. 10, no. 12, p. 237, 12 2022.

[19] Y. Wang, G. Andreeva, and B. Martin-Barragan, "Machine learning approaches to forecasting cryptocurrency volatility: Considering internal and external determinants," *International Review of Financial Analysis*, vol. 90, p. 102914, 11 2023.

[20] B. Christensen and N. Prabhala, "The relation between implied and realized volatility," *Journal of Financial Economics*, vol. 50, no. 2, pp. 125–150, 11 1998.

[21] L. Canina and S. Figlewski, "The Informational Content of Implied Volatility," *Review of Financial Studies*, vol. 6, no. 3, pp. 659–681, 7 1993.

[22] R. Engle and A. Patton, "What good is a volatility model?" *Quantitative Finance*, vol. 1, no. 2, pp. 237–245, 2 2001.

[23] M. McAleer and M. C. Medeiros, "Realized Volatility: A Review," *Econometric Reviews*, vol. 27, no. 1-3, pp. 10–45, 2 2008.

[24] A. P. Fassas and C. Siriopoulos, "Implied volatility indices – A review," *The Quarterly Review of Economics and Finance*, vol. 79, pp. 303–329, 2 2021.

[25] J. W. Taylor, "Volatility forecasting with smooth transition exponential smoothing," *International Journal of Forecasting*, vol. 20, no. 2, pp. 273–286, 4 2004.

[26] L. T. Hoang and D. G. Baur, "Forecasting bitcoin volatility: Evidence from the options market," *Journal of Futures Markets*, vol. 40, no. 10, pp. 1584–1602, 10 2020.

[27] Z. Shen, Q. Wan, and D. J. Leatham, "Bitcoin return volatility forecasting: A comparative study between GARCH and RNN," *Journal of Risk and Financial Management*, vol. 14, no. 7, p. 337, 2021.