A Survey on Econometric and Deep Neural Network Timeseries Forecasting Techniques

Christian Dominic Ataiza
University of Adelaide

Abstract

This research paper is a survey of both traditionally used econometric techniques and modern deep learning implementations, namely, Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) family of models (GARCH, EGARCH), Long Term-Short Term Memory (LSTM), and Recurrent Neural Networks (RNNs) for forecasting stock price volatility. The study researched on 4 specific index stocks, representative of their nation's market sentiments, namely S&P500 (GSPC), Dow Jones Industrial Average (DFI), EuroNext100 (NI100), and the Nikkei225 (N225). Data was scraped from yahoo!Finance. The historical close *price of the companies were considered, and using the* close *price, the paper was able to derive the stock's historical volatility. The objective of the paper is to compare the performance of GARCH-family models and the Deep Neural network on forecasting for the stock's future volatility on a per stock basis and create generalized conclusions by aggregating the model metrics. The paper was able to provide meaningful insight in the comparison between traditional econometric approaches as compared to machine learning approaches.*

1. Introduction

Stock price analysis and prediction have always been a point of interest for researchers, analysts, and investors. The ability to accurately forecast the future values of equities continues to be a challenge given the volatility and risk present in the market. Given the trillions of dollars flowing in and out of the market, there is great significance in strategies or techniques for accurate stock price prediction in a reasonable amount of time with a certain degree of risk tolerance. This research plans to investigate leading econometric techniques for time-series forecasting stock price volatility - generalized autoregressive conditional heteroskedasticity (GARCH), with modern deep learning techniques. Modern financial analysts rely on econometric forecasting tools such as GARCH and Autoregressive integrated moving average (ARIMA) due to their reliability in predicting time series data based on their capability to reduce the data into its constituent parts such as seasonality, residuals, and auto-correlation. However, due to the rapid advancement of modern machine learning, deep learning techniques are used for time-series forecasting such as through deep neural networks. This study aims to survey modern techniques namely time series analysis and forecasting using the GARCH family of models, Long Term-Short Term Memory (LSTM), and Recurrent Neural Networks (RNNs) to forecast the price volatility of multiple index stocks which tend to be representative of overall market sentiments at that time.

2. Related Work

Stock price prediction has been a problem that analysts and investors have been trying to solve for the longest time. It has been postulated that the stock market follows the Efficient Market Theory [9] where all public information regarding the equity, such as newly discovered information, trends and seasonality will be included in current price valuation at the moment of analysis. There are several ways to predict the future values of stock such as through technical analysis [7], time-series forecasting techniques namely forms of ARIMA [6], and forms of GARCH, and currently machine learning techniques.

In 2023, Amirashi created a study that compared GARCH-family models and forms of RNNs for forecasting cryptocurrency, and has shown promise for both forms of forecasting [1], and provided a methodology to combine both techniques to forecast volatility. Moreover, other literature such as Mustapa and Ismail's research, has supported that GARCH(1,1) is the best performing with generalization with stock price prediction. [8]

The stock prediction problem is well-documented as still today, continuous research is still being poured into this field.

3. Methodology

This study focused on two traditional time series models, GARCH(p,q) [2] and EGARCH(p,q) [10], as well as two

neural network architectures, RNNs [5] and LSTMs [4] to model stock price volatility.

3.1. Time-series Models

The paper explores two GARCH-type models as the representative of time series models namely GARCH and EGARCH(p,q). "The conditional variance equations of these models are described in Eqs.1,2. In each equation, ${\hat{\sigma_t}}^2$ refers to the conditional variance of the stock price returns - a function of $\mathcal{P}_t$ and $\mathcal{P}_t + 1$ to be later derived in the paper where $\mathcal{P}$ is the Close Price of the stock at time t. The return series is defined by the equation $\hat{r_t} = \hat{\mu_t} + \epsilon_t$ with $\hat{\mu_t}$ as the conditional mean of the return series. $\epsilon_t = \hat{\sigma_t} z_t$ is the heteroscedastic error term, and z_t is a random variable with a mean of zero and a unit variance. The parameters to be estimated from the maximization of the sample log-likelihood function are $w, \alpha_i, \beta_i, \gamma_i$. The distribution of the return series depends on the distribution of the standardized residuals, i.e., z_t. Three different distributions that will be considered in this study are Normal, Student's, and Generalized Error Distribution (GED)." [1]

$$ {\hat{\sigma_t}}^2 = w + \sum_{i=1}^{p} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_t - j^2 \tag{1} $$

Eq.1 shows that the conditional variance of price returns depends on both the square of the errors as well as the lagged conditional variances. However, the GARCH model fails to capture the leverage effect of volatility, which is why EGARCH overcomes this limitation by adding new parameter, γ_i that captures the asymmetric impact of news with negative shocks.

$$ \log {\hat{\sigma_t}}^2 = w + \sum_{i=1}^{p} (\alpha_i \epsilon_{t-1} + \gamma_i (|\epsilon_{t-1}| - E|\epsilon_{t-1}|)) + \sum_{j=1}^{p} \beta_j \log \sigma_{t-j}^2 \tag{2} $$

Financial time series deviate from the Gaussian distribution in real life since they exhibit excessive skewness and kurtosis which necessitates some sort of normalization which will be discussed when calculating for stock price residuals.

3.2. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) [5] are a type of artificial neural network designed to work with sequential data such as time series data by utilizing 'memory' as the information from earlier time steps influence the current input and output.

This is represented mathematically as the following equation:

$$ h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t) \tag{3} $$

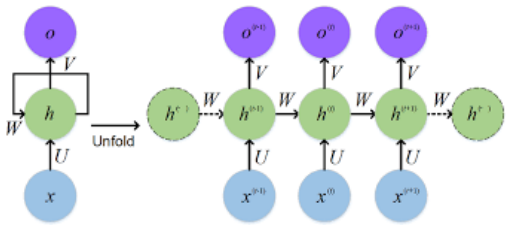

Figure 1. Unraveling RNN

where h_t is the hidden state at time step t, x_t is the input vector to enter a single unit of the RNN at time t, and W_{hh} and W_{xh} being the weights. However, as RNNs unravel in deep neural networks, it has the tendency to experience the vanishing gradient problem, where they stop learning new information due to the weights reaching infinitesimal small values. Additionally, the weight matrices will stay the same for all units of the RNN and across all time steps making it somewhat restrictive.

3.3. LSTM

Long Short-Term Memory (LSTM) is a type of RNN that was designed to overcome the limitations of traditional RNNs, particularly the vanishing gradient problem. LSTMs are used to process sequential data and are particularly effective for tasks that require an understanding of long-term dependencies, such as speech recognition or natural language processing. [4]

An LSTM cell is composed of several components. It contains a *memory cell* which is a unit that maintains a hidden state from one time step to the next; it is the key element of the LSTM mainly responsible for handling memory over time. A cell contains three types of gates namely the *input gate, forget gate, and output gate*. The input gate determines who much of the new input should be added to the memory cell. The forget gate decides which parts of the memory cell should be kept or discarded. Lastly, the output gate calculates the final output of the LSTM based on the updated memory cell. Mathematically, the LSTM are represented in the Eqs.

$$ f_t = \sigma_g(W_f x_t + U_f + h_{t-1} + b_f \tag{4} $$

$$ i_t = \sigma_g(W_i x_t + U_i + h_{t-1} + b_i \tag{5} $$

$$ o_t = \sigma_g(W_o x_t + U_o + h_{t-1} + b_o \tag{6} $$

$$ h_t = o_t * \sigma_c(c_t) \tag{7} $$

where f_t, i_t, o_t and h_t represents the forget gate, input gate, output gate and hidden state at time step t.

3.4. Model specifications

This paper focuses on both time-series data and types of various RNNs.

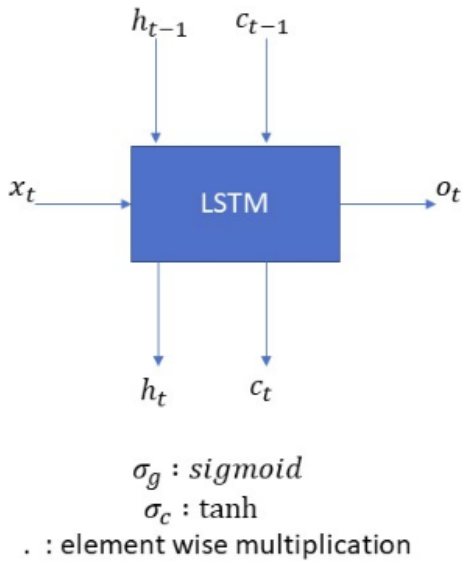

Figure 2. LSTM structure

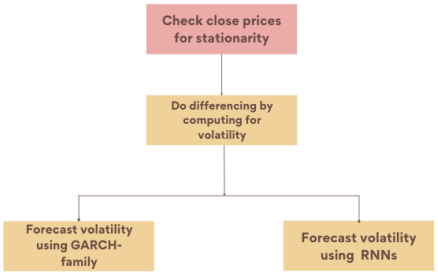

Figure 3. Flowchart to model steps in this experiment

For the time series data, literature has shown that GARCH(1,1) significantly outperforms other specifications of GARCH where p = 1 and q = 1 [8]. For this reason, EGARCH(1,1) will also be following similar specifications where p = 1 and q = 1, but as mentioned before, since the assumed distribution of the data significantly affects the model performance, three distributions were considered namely: Normal (Gaussian), Student's t, and General Error Distribution (GED). This will lead to a total of 6 model specifications.

For the RNNs, we explored specifically three counts of units in the RNN where units can be in [10,50,100]. Additionally, the paper explores two types of RNNs namely the vanilla RNN and LSTMs. This would also lead to a total of 6 model specifications.

4. Loss Function

Mean Squared Error (MSE) is commonly used as a loss function in various predictive models, including those applied to stock market forecasting. MSE penalizes larger deviations compared to smaller ones. In stock forecasting, where the magnitude of errors matters significantly, MSE helps emphasize and correct larger deviations between predicted and actual values. Additionally, MSE is a straightforward and computationally efficient metric, and compared to other functions, it is also a differentiable function, crucial for various optimization techniques like gradient descent. This property is advantageous when optimizing model parameters to minimize the loss function. Mathematically is expressed as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{8}$$

where Y_i is the actual value of the stock price volatility at observation i and $\hat{Y}_i$ is the forecasted value predicted by the model at observation i. An MSE close to 0 means that the model was able to make predictions close to the actual value.

5. Experiment and Results Analysis

5.1. Dataset

The dataset is index prices representative of the financial markets. The stocks were carefully chosen as they are the aggregate of the overall market and represent all the sectors inside a country's financial market. This study analysed the following: S&P500 (GSPC), Dow Jones Industrial Average (DFI), EuroNext100 (NI100), and the Nikkei225 (N225). These stocks were chosen to overall account for regional differences between the western and eastern sides of the world, while also accounting for the overall market sentiment present at those times for those specific markets.

The timeframe for the data was from 2018 to November 2023, to account for the price shocks due to COVID-19 from the period 2020 to 2022. Prices prior to 2018 were deemed not likely to cause further impact on current prices. For each index stock, daily stock price information was scraped from yahoo!Finance such as 'Date', 'Open', 'Close', 'High', and 'Low' represent the price the stock was at the start of the day, end of the day, and the highest price the stock reached for that day, and the lower price the stock reached for that day respectively. This study will mainly focus on the 'Close' column and derive computations from it as it represents the stock's price at the end of the day.

5.2. Calculating for Stock Price Volatility

Given that stock price tends to be auto-correlated [8], first-order differencing was conducted to make the data sta-

Index	Observations	Mean	SD	ADF-stat	ADF-p value
GSPC	1442	0.04257	0.02756	-5.055090	0.00002
DJI	1442	0.04010	0.02896	-4.48257	0.00021
N100	1470	0.03954	0.02133	-5.04313	0.00002
N225	1397	0.043892	0.01588	-4.7579	0.00007

Table 1. Descriptive Statistics of the HV_t values of each stock

tionary. This is needed since autocorrelation denotes dependence among the data earlier data, and this can cause errors in the forecast. Stationarity is necessary to make assumptions regarding our data when proven that the data are independent from each other. Additionally, this lines up nicely with what investors and financial analysts want when forecasting stock prices. The actual price itself is not as important as the trend of it going up or going down the next day. This study models the expected of the stocks as:

$$r_t = \log(P_t) - \log(P_{t-1}) \tag{9}$$

where r_t denotes the return of the stock at time t, and P_t and P_{t-1} denotes the close price of the stock at time t and $t-1$ respectively.

Note that the return values calculated are not necessarily equivalent to volatility. Given that volatility is an unempirical metric, this paper will use the standard deviation the index stock has during a certain window serving as a proxy for volatility. Therefore, historical volatility at time t is calculated using the Eq.:

$$HV_t = \sqrt{\frac{1}{T}\sum_{i=t-T}^{t}(r_i - \hat{r_t})^2} \tag{10}$$

where r_i is the price return of the index stock on day i, and $\hat{r_t}$ is the average return of index stock during past T trading days. This metric was adapted from Amirshahi's 2023 paper. [1]

Conveniently, computing for the price volatility squeezes our data to have a smaller range compared to having the original magnitude in price. This normalizes our data to squeeze in the range of being [-1,1] and significantly and consequently also increases the time it takes for the loss functions to converge. The descriptive statistics for the price volatility can be seen in Table 1. After all the transformations, it can be confirmed that the HV_t is stationary given the augmented Dickey-Fuller test (ADF) statistic. [3] For the purpose of this paper, the window length is kept constant where T = 30. This averages around a month's worth of data per window. Further exploration of different sizes will be considered for future experimentation.

5.3. Train-Validation-Test Split

The data was split into train, validation, and test sets respectively, with the cutoffs of the train data at the end of

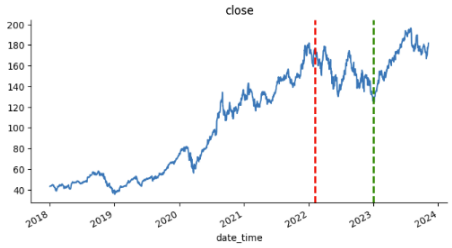

Figure 4. Showing the Train-validation-test split by date on the S&P500

Q2 2022 and validation cut off at January 2023. The test data ranges from January 2023 to November 2023. Effectively, this creates a 70-15-15 split. Shown in Fig.4 are the S&P500 prices divided where the training and validation cut-offs were.

5.4. Metrics

To measure the effectiveness of the models, two metrics were used to compare the different models. Given that we accurately capture the prediction accuracy, we optimized on the MSE as shown in Eq.8, the paper will compare the MSEs of the models. As mentioned in the equation above, the MSE emphasizes large deviations between the actual and predicted. A lower MSE indicates a better fit. However, due to the magnitude of the data being very small, another metric will be compared alongside MSE. The paper will also explore the Mean Absolute Percentage Error (MAPE) of the models. MAPE calculates the absolute percentage differences between the actual and predicted values given by the Eq.

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{A_i - F_i}{A_i}\right| \tag{11}$$

where A_i is the actual value for observation i and F_i is the forecasted value at observation i, and n is the total number of observations. Compared to MSEs which will be very small due to the magnitude of the data, MAPE can provide better context on how the model well correctly predicts the residuals as a percentage where a MAPE closer to 0 indicates a better performance.

5.5. Training Results

5.5.1 Time-series Forecasting Results

Table 2 summarizes the MSE and MAPE metrics of the time-series forecasting. It can be observed that the GARCH-normal models yield to the lowest MSE values excluding outliers in the N225 in predicting the test data. However, the best MAPE or the percentage of accuracy of the model is not necessarily forecasted by the GARCH-normal. For the Dow-Jones Index, EGARCH-GED had the

Table 2. Results of time-series models for HV_t prediction per index stock. T = 30

Model	Metric	GSPC	DJI	N100	N225
('GARCH', 'normal')	MSE	0.0000601	0.0000353	0.0000426	71.9203581
	MAPE	0.1726368	0.1953547	0.1512346	238.0923412
('GARCH', 't')	MSE	77.4308245	0.0023135	38.4829470	1.5638494
	MAPE	281.1678408	1.8038704	199.5812755	35.1431204
('GARCH', 'ged')	MSE	0.0000652	0.0527797	18.8288833	0.0050477
	MAPE	0.1752924	8.5169697	139.5924524	1.9574931
('EGARCH', 'normal')	MSE	0.0000635	0.0000431	0.0000548	0.0000365
	MAPE	0.1742110	0.2248224	0.1487933	0.1356205
('EGARCH', 't')	MSE	0.0001265	38,656,859,579,308.0000000	0.0000481	0.0000365
	MAPE	0.2470197	229,436,693.9346490	0.1442952	0.1356200
('EGARCH', 'ged')	MSE	0.0000739	0.0000357	0.0000549	0.0006158
	MAPE	0.1815505	0.1706125	0.1489462	0.7060422

Table 3. Average of the performance time-series models across the 4 stocks.

Model	Metric	Value
('GARCH', 'normal')	MSE	0.0000601
	MAPE	0.1726368
('GARCH', 't')	MSE	77.4308245
	MAPE	281.1678408
('GARCH', 'ged')	MSE	0.0000652
	MAPE	0.1752924
('EGARCH', 'normal')	MSE	0.0000635
	MAPE	0.174211
('EGARCH', 't')	MSE	0.0001265
	MAPE	0.2470197
('EGARCH', 'ged')	MSE	0.0000739
	MAPE	0.1815505

lowest MAPE, but with GARCH-normal close in terms of magnitude.

To account for the generalized performance of the model the metrics have been average across all 4 indices in Table 3.

On average, GARCH-normal performs the best among the different models with the MSE of the forecasted values 6.01×10^{-5} with an average MAPE of 17.26%. Additionally, it can be observed that the student's t distribution also performed the worst when forecasting the stocks on average in both GARCH and EGARCH models respectively. However, there is not enough conclusive evidence to show that GARCH outperforms EGARCH overall for stock price volatility prediction.

5.5.2 Deep Learning Results

Table 5 shows that the models performed their predictions to an acceptable degree comparable to the time series forecasting. The top model predicting the EuroNet100 significantly

Table 4. Results of RNN models for HV_t prediction per index stock. T = 30

Index	Model	MAPE	MSE
N100	RNN_100	0.14412278	0.00003874
N100	LSTM_100	0.19909834	0.00004566
GSPC	RNN_100	0.17890725	0.00004856
N100	LSTM_50	0.27574032	0.00007749
GSPC	LSTM_50	0.26758101	0.00007942
N100	RNN_50	0.28816802	0.00008455
N225	RNN_10	0.24256944	0.00008468
N225	LSTM_10	0.24512370	0.00008673
N225	LSTM_100	0.24635235	0.00008745
GSPC	LSTM_10	0.28972605	0.00009339
N225	RNN_50	0.25741527	0.00009462
N225	LSTM_50	0.26074540	0.00009629
N100	LSTM_10	0.32053208	0.00010510
GSPC	LSTM_100	0.32514447	0.00011528
N100	RNN_10	0.33765883	0.00011662
GSPC	RNN_10	0.35003719	0.00013208
GSPC	RNN_50	0.35685990	0.00013622
N225	RNN_100	0.34783710	0.00016007
DJI	LSTM_10	0.48243395	0.00017543
DJI	LSTM_50	0.50714647	0.00019259
DJI	LSTM_100	0.51061257	0.00019515
DJI	RNN_50	0.53854746	0.00021486
DJI	RNN_10	0.68001387	0.00033760
DJI	RNN_100	0.81013643	0.00046928

outperformed any model from the time-series forecasting with an MSE of 3.87×10^{-5} and MAPE of 14.41%. On a per-stock basis, it can be observed the RNNs have outperformed their time-series contemporaries when comparing only the best models. Likewise, to draw inferences about generalizability, the metrics were also summarized by averaging the values of the MSE and MAPE to see the average

Table 5. Average of the performance RNN models across the 4 stocks.

Model	MAPE	MSE
LSTM_100	0.32030193	0.00011089
LSTM_50	0.32780330	0.00011145
LSTM_10	0.33445395	0.00011516
RNN_50	0.36024766	0.00013256
RNN_10	0.40256983	0.00016774
RNN_100	0.37025089	0.00017916

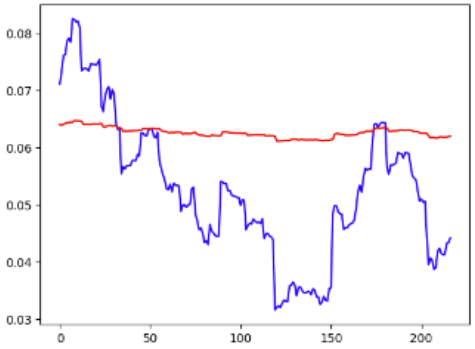

Figure 5. LSTM predictions of S&P500 on test; Units = 100

performance of the models across the 4 indices in order to the different types of RNNs with each other as shown in Table 5.

On average, it is clearly observed that the RNN models performed worse than LSTM across all unit numbers. This means that there is clear improvement for choosing LSTM over RNNs in stock forecasting. It can be hypothesized that 100 units might not be enough units to capture the model and the possibility of adding more layers can increase the model performance. However, compare to the average performance of GARCH-normal, the LSTM-100 performed somewhat worse at a MSE of 1.10×10^{-4} and a MAPE of 32.03%, but the deep learning models did not have major outliers in its predictions compared to the time-series techniques. Figures 5,6 show the difference in the stock predictions across LSTM100 and GARCH-Normal in the S&P500 data.

5.5.3 Deep Learning Loss Curves

The models all optimized for minimizing the residual values. It can be seen in the figure that, the data was able to get low MSE for both training and validation curves, and moreover, the models have all converged relatively quickly.

Figure 6. GARCH-normal predictions of S&P500 on test

6. Code

The code for this project can be found on this link: https://github.com/nicoataiza/CS7318/blob/main/Assignment_3_loops.ipynb

7. Conclusion

The study surveys forecasting techniques predominantly used by most financial analysts namely GARCH and the GARCH family of models and modern deep learning neural networks. On the general cases, GARCH assuming a normal distribution outperforms the generalized cases of LSTMs and RNNs in performing stock price volatility in terms of MSE, but our survey also showed that it was the deep neural network which had the best prediction at a individual stock level. Additionally, the study showed that for the case of stock price volatility prediction, LSTMs greatly outperform the RNNs on average. However, at best, both time-series and LSTM models only have around 17% MAPE given the stock's heavy tailed distribution. Further research should be considered on checking the performance on stock with other forms of sequential neural networks such as with transformers or possible graph neural networks. Additionally, combinations or hybrids of the two models can be explored further.

References

[1] Bahareh Amirshahi and Salim Lahmiri. Hybrid deep learning and garch-family models for forecasting volatility of cryptocurrencies. *Machine Learning with Applications*, 12:100465, 2023. 1, 2, 4

[2] Robert F. Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica*, 50(4):987–1007, 1982. 1

[3] R.I.D. Harris. Testing for unit roots using the augmented dickey-fuller test: Some issues relating to the size, power and

the lag structure of the test. *Economics Letters*, 38(4):381–386, 1992. 4

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[5] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982. 2

[6] Rob J Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice, 2nd edition*. OTexts, 2018. 1

[7] CFA Institute, 2023. Last accessed 22 November 2023. 1

[8] Farah Hayati Mustapa and Mohd Tahir Ismail. Modelling and forecasting samp;p 500 stock prices using hybrid arima-garch model. *Journal of Physics: Conference Series*, 1366(1):012130, nov 2019. 1, 3

[9] NASDAQ. Last accessed 22 November 2023. 1

[10] Daniel B. Nelson. Conditional heteroskedasticity in asset returns: A new approach. *Econometrica*, 59(2):347–370, 1991. 1

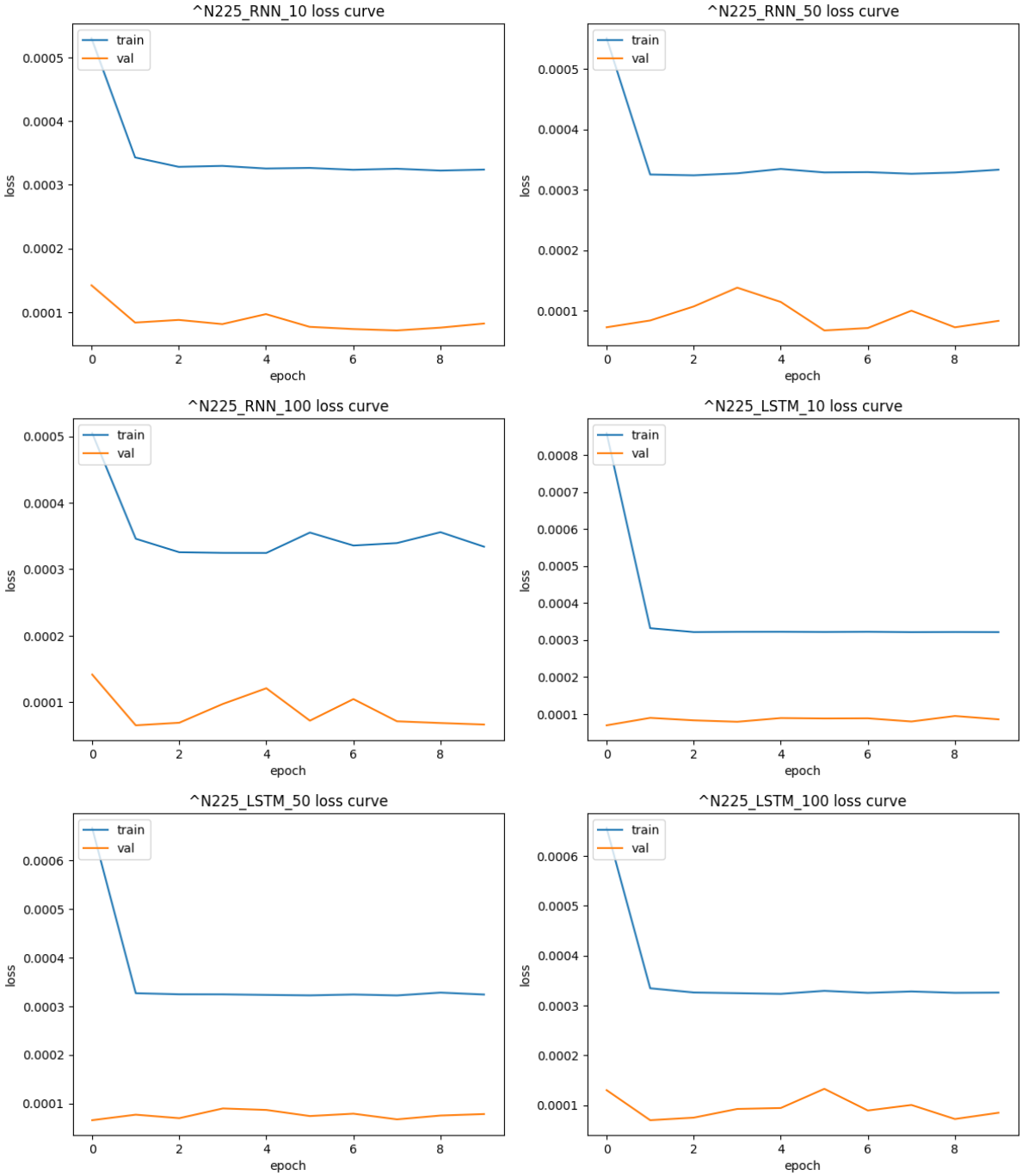

Figure 7. Training and Validation loss curves for sample stock = N225. The title of the plot indicates what RNN is being used and what is the number of units e.g. LSTM_100 is a LSTM using 100 units.