Journal of Quality Measurement and Analysis *JQMA* 19(2) 2023, 81-97
e-ISSN: 2600-8602 *http://www.ukm.my/jqma*

FINANCIAL MARKET PREDICTIONS WITH DEEP LEARNING

(Ramalan Pasaran Kewangan dengan Pembelajaran Mendalam)

YAP ZHONG JING & DHARINI PATHMANATHAN*

ABSTRACT

Forecasting the financial market has proven to be a challenging task due to high volatility. However, with the growing involvement of computational methods in econometrics, models built with deep learning neural networks have been more accurate in capturing the dynamics of financial market data compared to the commonly used time series models such as the ARIMA and GARCH models. In this study, four deep learning models were applied to eight separate investments, namely stocks (AAPL, TSLA, ROKU, BAC), currency exchange rates (GBP/USD and USD/SEK) and exchange-traded funds (SQQQ and SPXS) to compare their forecasting abilities. The four deep learning models consists of three recurrent neural networks (RNN) which are the vanilla recurrent network (VRNN), long short-term memory (LSTM) and gated recurrent units (GRU), along with the convolutional neural networks (CNN). The models were tuned to be time efficient and evaluated with RMSE and MAPE. Results show that GRU was the overall best model, with exceptions to the LSTM performing better with the exchange traded funds.

Keywords: CNN; financial market; GRU; LSTM; RNN

ABSTRAK

Meramal pasaran kewangan telah terbukti sebagai tugas yang mencabar disebabkan oleh ketidastabilan yang tinggi. Walau bagaimanapun, dengan penglibatan yang semakin meningkat dalam kaedah pengiraan dalam ekonometrik, model yang dibina dengan rangkaian neural pembelajaran dalam adalah lebih tepat dalam menangkap dinamik data pasaran kewangan berbanding model siri masa yang biasa digunakan seperti model ARIMA dan GARCH. Dalam kajian ini, empat model pembelajaran dalam telah digunakan pada lapan pelaburan berasingan, iaitu saham (AAPL, TSLA, ROKU, BAC), kadar pertukaran mata wang (GBP/USD dan USD/SEK) dan dana yang didagangkan di bursa saham (SQQQ dan SPXS) untuk membandingkan kebolehan ramalan mereka. Empat model pembelajaran dalam terdiri daripada tiga rangkaian neural berulang (RNN) iaitu rangkaian neural biasa (VRNN), long short-term memory (LSTM) dan gated recurrent units (GRU), bersama-sama dengan rangkaian neural konvolusi (CNN). Model-model ini telah diselaraskan untuk menjadi cekap dari segi masa dan dinilai dengan RMSE dan MAPE. Keputusan menunjukkan bahawa GRU adalah model terbaik secara keseluruhan, dengan pengecualian LSTM yang lebih baik dalam dagangan dana yang didagangkan di bursa saham.

Kata kunci: CNN; pasaran kewangan; GRU; LSTM; RNN

1. Introduction

Financial market predictions have become an area of interest among institutions surrounding the computational finance field. The financial market undoubtedly reflects the state of economic growth globally, hence analysing it to make accurate predictions into its future would be helpful across multiple disciplines that are dependent on the financial market. Time series forecasting is a means of drawing inferences from a time series and making the appropriate forecast into

the future by using a suitable theoretical model to fit onto the time series (Granger & Newbold 1986).

One of the most widely known models to be used to model financial market prices is the autoregressive integrated moving average (ARIMA) model which can convert non-stationary data to stationary data through differencing. It involves linear forecasting equations and is efficient in forecasting short-term financial market movements which leverages past time series data to forecast future trends. Subsequently, non-linear models such as the generalised autoregressive conditional heteroskedasticity (GARCH) model was introduced to accommodate the dynamic fluctuations of the data by modelling its variance which could better handle volatile data (Sparks & Yurova 2006).

Although these statistical models are effective in forecasting a short-term time frame, due to the highly complex and dynamic nature of financial market data, these models tend to fail in capturing the hidden dynamics within the data which calls for more robust forecasting models.

With the rise of artificial intelligence (AI) applications in finance, forecasting practices have also caught on by taking advantage of the potential of machine learning with deep learning models which have displayed incredible success in a variety of applications such as speech recognition, medical diagnostics, gameplay, and natural language processing. Deep learning neural networks are capable of learning mappings and relations that are arbitrarily complex, and when applied to time series data, can process, and recognise complex dependencies within the data, and therefore can undertake the challenge of modelling the highly unpredictable and dynamic stock market data.

The recurrent neural networks (RNN) are a deep learning neural network that is commonly used on sequential or temporal data. It is known for its "memory" component where the predicted outputs are influenced by not only the current input, but also the previous inputs by a process called backpropagation through time. The Vanilla RNN (VRNN), long short-term memory (LSTM), and the gated recurrent unit (GRU) are the most used RNNs in the financial market scene. VRNN is the most basic form of an RNN which contains only a single unit of "memory", causing it to overload easily, leading to gradient vanishing or exploding issues, especially when encountering larger volumes of data. To overcome this, LSTM was introduced to address these error backflow issues (Hochreiter & Schmidhuber 1997). With the addition of memory cells and gates, LSTM can store information into its "long-term memory" and "short-term memory". GRU was later introduced to address the gradient issue with an encoder-decoder approach (Cho *et al*. 2014).

Another deep learning neural network that is commonly used is a type of feed forward network known as convolutional neural network (CNN) which was first introduced for the purpose of pattern recognition whose networks were inspired by the organization of neurons in an animal cortex (LeCun *et al*. 1998). CNNs use the convolution and pooling layers within its network to learn information directly from the raw training data. Although CNNs are not able to process sequential data such as the stock market data directly, when presented in a grid-like configuration, the CNN can also be used for time series analysis.

This work aims to compare the predictive performances of four different deep learning neural networks: VRNN, LSTM, GRU, and CNN to determine the most accurate and time efficient model for forecasting the closing prices of stocks and foreign currency pairs.

2. Literature Review

Stock price prediction is one of the most studied financial time series applications (Ozbayoglu *et al*. 2020) and over the years, several models have been developed for stock market time series

data for forecasting purposes. The two most widely used techniques for time series forecasting are statistical and computational methods (Preeti *et al*. 2018).

The statistical methods are those that are used in econometrics for price forecasting studies which are traditionally performed using the Box and Jenkins' ARIMA model. Dong *et al*. (2017) used ARIMA to forecast Apple's stock price and noted its strong potential when it came to short-run forecasts, whereas Ariyo *et al*. (2014) also came to the same conclusion when using the ARIMA model to forecast prices of stocks from NYSE and NSE. However, the ARIMA model's weakness is its assumption of constant variance in errors, desensitizing it from capturing more non-linear activities within the data which can be overcome by using the GARCH model. This is evident in Sparks and Yurova's (2006) where the forecasting abilities of ARIMA and GARCH models were compared on 31 separate company stocks indicating that the ARIMA model did not provide accurate estimates. Then, hybrid models between the two models emerged similar to Babu and Reddy's (2014) where they document their experiments of comparing a hybrid ARCH-GARCH model with the standalone models using Indian stocks. Results showed that the hybrid model outperforms the standalone models with a wide margin. Another model that was compared in this article was an artificial neural network that managed to surpass the hybrid ARCH-GARCH model.

Recently, computational methods have been on the rise and have totally changed the way we tackle the topic of modelling, with concerns of the lower involvement of proper statistics being used in modelling data and forecasting (Nielsen 2019). However, many promising literatures have shown the superiority of these computational neural networks over traditional statistical methods. Siami-Namini *et al*. (2018) showed that LSTM's average reduction of error rates was much lower compared to that of the ARIMA model.

Hiransha *et al*. (2018) compared four deep learning architectures that is the multilayer perceptron (MLP), RNN, LSTM and CNN with the ARIMA model, using a 10-year span of day-wise closing prices of different stocks from NYSE and NSE. Results showed that the ARIMA model was unable to grasp the nonlinear variation of the data, concluding that deep learning models outperform the ARIMA model. In the proposed work, it was also reported that CNN was more accurate compared to its three other deep learning counterparts. Selvin *et al*. (2017) also used the same four deep learning models with ARIMA and obtained similar results. Kumar *et al*. (2021) proposed a stacked LSTM model to be compared against moving average (MA) and XGBoost models which were implemented on the stock price of Infosys Limited and evaluated using RMSE and MAPE which indicated that the LSTM model outperformed the other two benchmark methods.

There is much literature that compares RNN's predictive accuracy. The most frequently used ones in the context of stock price predictions are the VRNN, LSTM and GRU models. For example, Saud and Shakya (2020) compared the performances of the three different recurrent neural networks and conducted multivariate analysis using six other features other than the closing price to forecast the next day closing price. The data used was the stock of two commercial banks listed on the Nepal stock exchange (NEPSE) and evaluated using MAPE with results showing that both LSTM and GRU are better than the VRNN model with the GRU model performing slightly better than the LSTM model. Mohammad and Owda (2021) on the other hand used GRU, LSTM and a bi-directional LSTM to carry out forecasting on the prices of cryptocurrencies. The GRU model had the lowest MAPE compared to the other two models, making it the best model among the three, however, it was noted that the LSTM and bi-directional LSTM models were also reliable in producing satisfactory predicted results. Sako *et al*. (2022) used VRNN, LSTM and GRU models to forecast the closing prices of eight different

stock market price index and six different currency exchange rates related to the USD and concluded that GRU is the overall best model.

Bringing CNN into the picture, results from the works of Selvin *et al*. (2017) and Hirnansha *et al*. (2018) show that it can outperform the other frameworks of RNN well. Di Persio and Honchar (2016) compared the performances of MLP, CNN and LSTM in forecasting the future price of the S&P500 index data from 1950 to 2016 for their experiment with various neural networks. They used opening price, closing price, high volume and low volume data as inputs and MSE for model evaluation. They have also included a novel wavelet-CNN model which incorporates wavelet neural networks (WNN) into CNN for price prediction. In the presented results, it seems like all the deep learning models are fairly accurate with CNN having the least MSE. Jain *et al*. (2018) implemented the CNN and LSTM on the stock prices of two different companies and found that CNN is more accurate compared to LSTM.

Taking motivation from the success of deep learning neural networks over classical statistical methods in the reviewed literature above, this work will serve to compare the forecasting abilities of the VRNN, LSTM, GRU, CNN models for univariate analysis on the closing prices of various investments.

3. Data Description

Five years' worth of historical data dated from 1 January 2017 to 31 December 2021 of daily closing prices of different equity stocks, exchange traded funds (ETF) and currencies were retrieved from Yahoo! Finance. These investment data are:

(1) Apple Inc (AAPL)
(2) Tesla Inc (TSLA)
(3) Roku Inc (ROKU)
(4) Bank of America Corp (BAC)
(5) GBP/USD
(6) USD/SEK
(7) ProShares UltraPro Short QQQ ETF (SQQQ)
(8) Direxion Daily S&P 500 Bear 3X Shares ETF (SPXS)

The datasets are then split into training and testing sets where the training set consists of the first 80% of the data, and the remaining 20% will be the testing set.

4. Methodology

In this study, univariate forecasts were performed using four different types of deep learning models which are Vanilla Recurrent Neural Networks (VRNN), Long-Short Term Memory (LSTM), Gated Recurrent Units (GRU), and Convolutional Neural Networks (CNN). Computational operations were carried out using Google Colab which runs on Python 3.10.11 along with the open source library Keras (Chollet 2015) to build the deep learning models.

4.1. *Vanilla Recurrent Neural Networks (VRNN)*

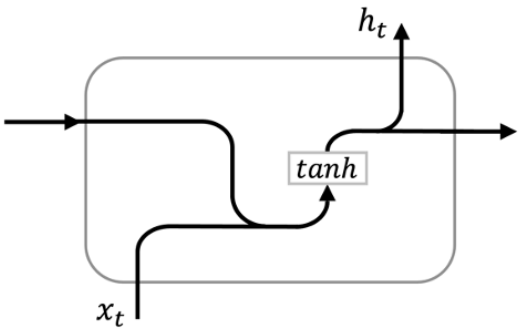

Figure 1: VRNN cell architecture

The VRNN is the most basic form of a recurrent neural network where its connections between the cell structure can accept variable inputs and outputs. VRNNs can store a lot of past information efficiently and use their internal memory to process input sequences and are purely deterministic. The operations of a VRNN can be expressed mathematically as follows:

$$h_t = \tanh\left(Wh_{t-1} + Ux_t \right) \tag{1}$$

where x_t is the input variable at time t, h_t is the hidden state output gate, and the weight parameters W and U of the input and the hidden state respectively.

The only concern of a VRNN is that the gradient of the loss function decays exponentially with time which might interrupt the learning process of the model. This problem is called the vanishing gradient problem. The architecture of the VRNN model used is depicted below:

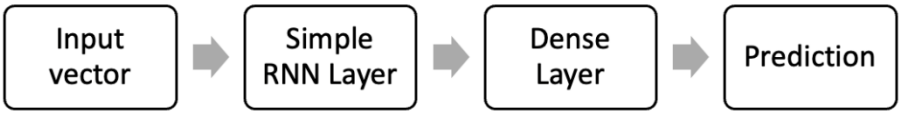

Figure 2: VRNN model architecture

4.2. *Long-Short Term Memory (LSTM)*

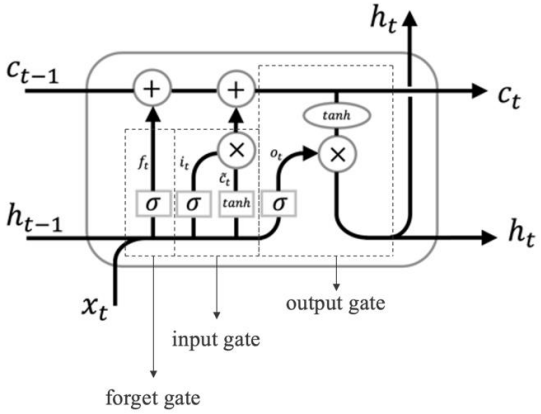

Figure 3: LSTM cell architecture

LSTM is a special type of RNN and as its name suggests, possesses the long-term and short-term memory. Its cell architecture is composed of gates to regulate information to be kept or discarded before relaying the information to the next cell. These gates are the input gate, forget gate, and output gate. The input gate filters information consisting of short-term memory and current input and decides what is to be stored in its long-term memory. Then, at the forget gate, the LSTM cell then decides what to discard from its long-term memory. Finally, along with the current input and the updated long-term short-term memory, it generates an output at the output gate to be passed on to the next cell. The operations of the LSTM cell can be mathematically expressed as follows (Hossain *et al*. 2018):

$$i_t = \sigma\left(W^{(i)} x_t + U^{(i)} h_{t-1} + b^{(i)} \right) \quad (2)$$

$$f_t = \sigma\left(W^{(f)} x_t + U^{(f)} h_{t-1} + b^{(f)} \right) \quad (3)$$

$$o_t = \sigma\left(W^{(o)} x_t + U^{(o)} h_{t-1} + b^{(o)} \right) \quad (4)$$

$$u_t = \tanh\left(W^{(u)} x_t + U^{(u)} h_{t-1} + b^{(u)} \right) \quad (5)$$

$$c_t = i_t \odot u_t + f_t \odot c_{t-1} \quad (6)$$

$$h_t = o_t \odot \tanh\left(c_t \right) \quad (7)$$

From the above equations, for an input vector the LSTM unit at time step t: i_t is an input gate, f_t the forget gate, o_t is an output gate, c_t is a memory cell, u_t is an activation function, the hidden state h_t, the weight parameters W and U of the input and the hidden state respectively and b the bias.

The architecture of the LSTM model used is depicted below:

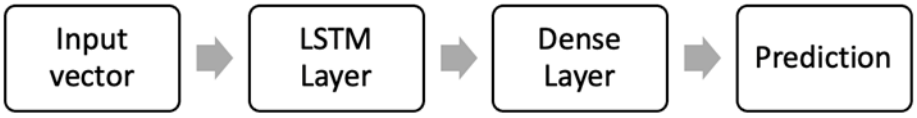

Figure 4: LSTM model architecture

4.3. *Gated Recurrent Units (GRU)*

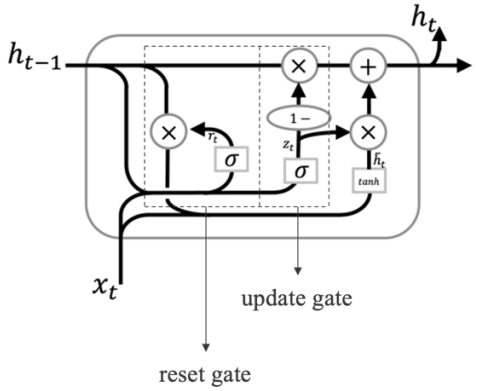

Figure 5: GRU cell architecture

GRU is also a special variation of an RNN which closely resembles LSTM but with a simplified cell structure. Its cell structure contains two gates, the update gate, and the reset gate. The update gate decides how much past information needs to be passed on to the next state, while the reset gate decides how much past information needs to be neglected. The operations of GRU can be mathematically represented as follow (Hossain *et al*. 2018):

$$z_t = \sigma\left(W^{(z)} x_t + U^{(z)} h_{t-1} + b^{(z)} \right) \tag{8}$$

$$r_t = \sigma\left(W^{(r)} x_t + U^{(r)} h_{t-1} + b^{(r)} \right) \tag{9}$$

$$a_t = \tanh\left(W x_t + r_t \odot U h_{t-1} + b^{(h)} \right) \tag{10}$$

$$h_t = z_t \odot h_{t-1} + \left(1 - z_t \right) \odot a_t \tag{11}$$

From the above equations, z_t is the update gate, r_t is the reset gate, a_t is the activation function, h_t is the hidden state output gate, and the weight parameters W and U of the input and the hidden state respectively and b the bias.

With fewer parameters than LSTM, the GRU model is expected to be able to train faster. The architecture of the GRU model used is depicted below:

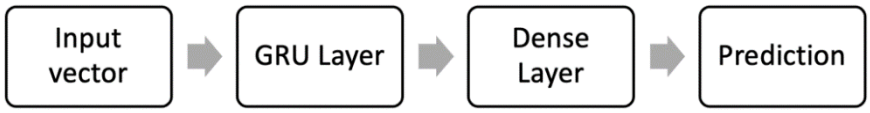

Figure 6: GRU model architecture

4.4. *Convolutional Neural Networks (CNN)*

CNN, unlike the previous models is a different class of feed-forward neural network which processes data that has a grid or matrix-like topology. Therefore, a time series data can be thought of as a 2D grid of pixels. The architecture of the CNN typically inputs data into a convolutional layer where convolution is applied to extract and filter out information from the input, followed by pooling layers which performs down sampling to compress the data, and full connection layers which learns the pooled information before producing an output. The architecture of the CNN model used is depicted below:

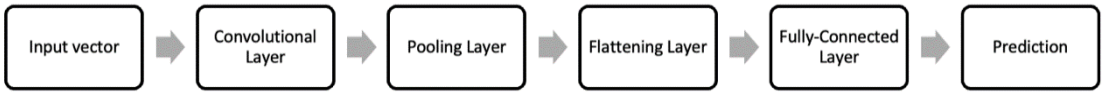

Figure 7: CNN cell architecture

5. Setting Up the Models

All models in this study will be equipped with the Adam optimizer and the mean squared error loss function. An optimizer is a function that modifies the attributes of the model such as the weight. The Adam optimizer is computationally efficient, has little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data (Kingma & Ba 2014). The MSE loss function on the other hand can be used to compute the mean squared error between the predictions and the test data.

The epoch number, which is the number of times the model goes through the dataset is set at three for all models to reduce time taken to complete the back propagation process of the

recurrent neural networks which may be time consuming seeing how years' worth of closing price data is being used. Layers within all the models are also subjected to fine tuning.

6. Evaluation Metrics

The models will then be evaluated using root mean squared error (RMSE) and mean absolute percentage error (MAPE). MAPE takes the absolute percentage of error between the predicted values and actual values and later averaging it over all predictions, while RMSE calculates the average magnitude of the error between predicted and actual values. The equations representing the error metrics are presented below:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n} \left| y_t - \widehat{y_t} \right|}{n}} \tag{12}$$

$$MAPE = \frac{100\%}{n} \sum_{t}^{n} \left| \frac{y_t - \widehat{y_t}}{y_t} \right| \tag{13}$$

where y_t is the actual value and $\widehat{y_t}$ is the predicted value of the *t*th observation.

7. Results and Discussion

7.1. *Results*

The experiment was conducted on four separate deep learning models, namely the VRNN, GRU, LSTM and CNN models, which were fed with a variety of stock market data that display different dynamics, behaviour, and volatility across a five-year time span from 1st January 2017 to 31st December 2021. The RMSE, MAPE and execution time are tabulated below in Table 1 and Table 2 respectively.

Table 1: RMSE & MAPE

	RMSE				MAPE			
	VRNN	LSTM	GRU	CNN	VRNN	LSTM	GRU	CNN
AAPL	2.754	3.158	**2.250**	5.488	0.016	0.018	**0.012**	0.033
TSLA	12.782	15.875	**9.698**	20.933	0.032	0.045	**0.025**	0.061
ROKU	16.257	14.962	**12.373**	30.626	0.038	0.034	**0.028**	0.073
BAC	0.814	0.748	**0.734**	0.926	0.016	0.015	**0.015**	0.018
GBP/USD	0.007	0.007	**0.006**	0.009	0.004	0.004	**0.003**	0.005
USD/SEK	0.057	0.054	**0.048**	0.076	0.005	0.005	**0.004**	0.007
SQQQ	12.688	**5.358**	18.616	24.951	0.273	**0.106**	0.405	0.549
SPXS	0.867	**0.863**	2.030	16.240	0.024	**0.024**	0.079	0.693

After splitting the dataset, it will then be transformed by normalising the data with min-max normalisation so that values are rescaled into ranges between 0 and 1 to facilitate model learning

which effectively reduces training time. The output of predictions will later be rescaled to reflect the actual closing prices predicted.

The normalized training set is then transformed into a supervised learning configuration that is compatible as a deep learning model input vector. The window size is fixed at 60, which is the number of observations granted to our model to make the next prediction, so that it is sufficient for our model to learn longer dependencies.

Experiments were carried out on Google Colab which provided an Intel Xeon CPU with 2.20GHz and 12 GB of RAM with limited access to high-memory VMs which have 25 GB RAM.

Table 2: Execution time of each deep learning models on different investments

	Execution Time (seconds)			
	VRNN	LSTM	GRU	CNN
AAPL	55.876	95.985	95.120	**17.103**
TSLA	63.439	104.694	63.834	**9.872**
ROKU	69.545	89.064	70.549	**7.159**
BAC	70.420	104.236	114.174	**35.685**
GBP/USD	70.361	149.714	98.623	**9.670**
USD/SEK	83.094	77.368	101.136	**7.107**
SQQQ	71.745	110.765	113.486	**10.838**
SPXS	81.194	124.715	103.478	**12.672**

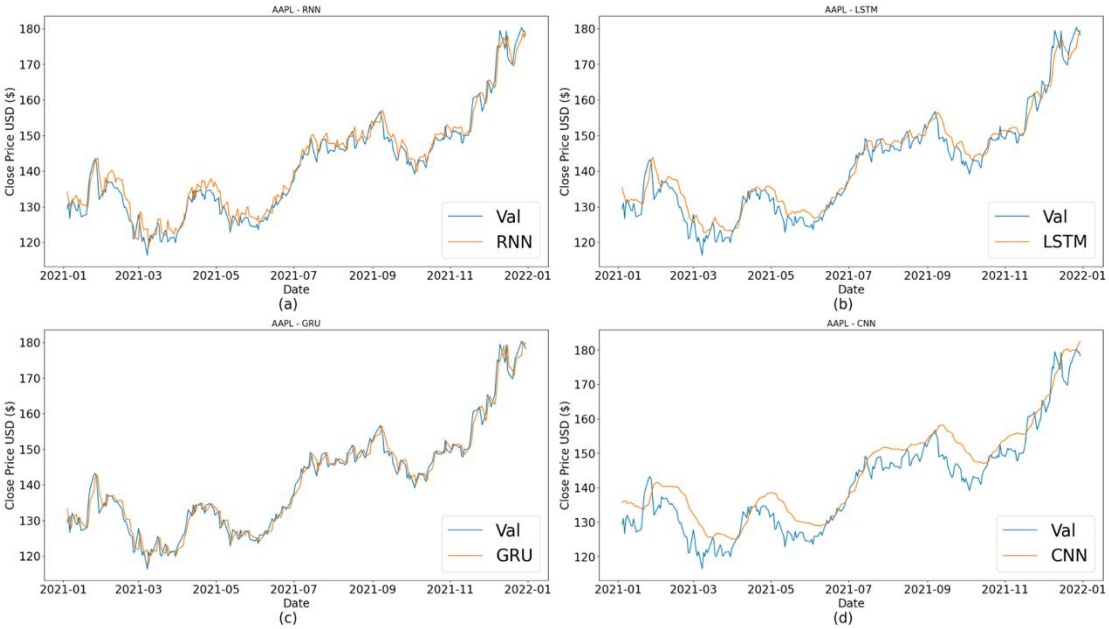

Figure 8: AAPL prediction vs actual closing price using (a) VRNN; (b) LSTM; (c) GRU; (d) CNN

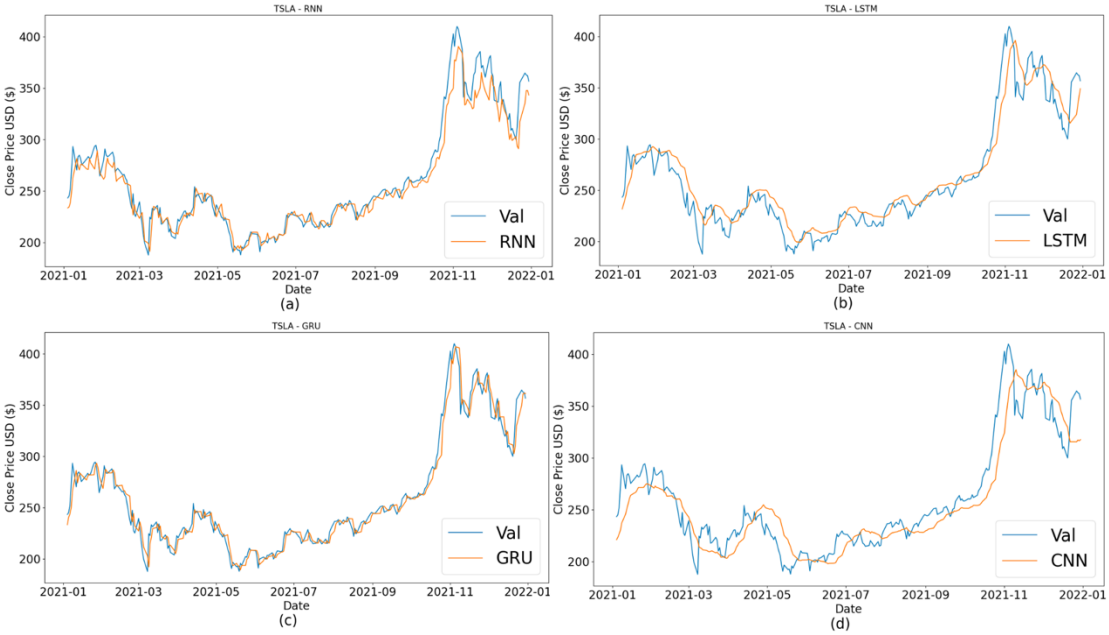

Figure 9: TSLA prediction vs actual closing price using (a) VRNN; (b) LSTM; (c) GRU; (d) CNN

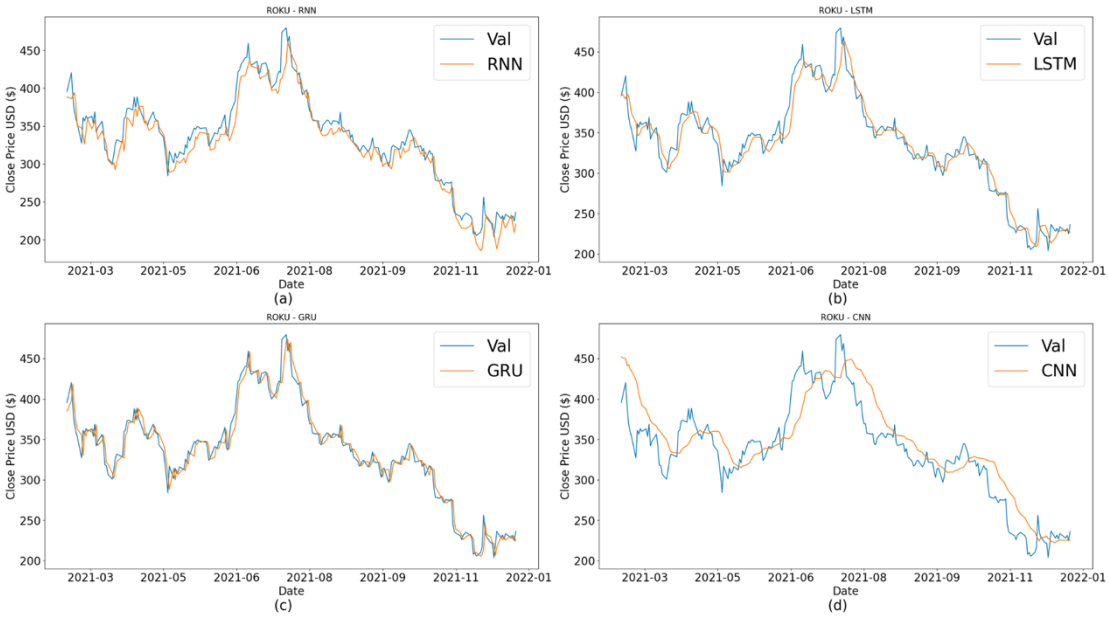

Figure 10: ROKU prediction vs actual closing price using (a) VRNN; (b) LSTM; (c) GRU; (d) CNN

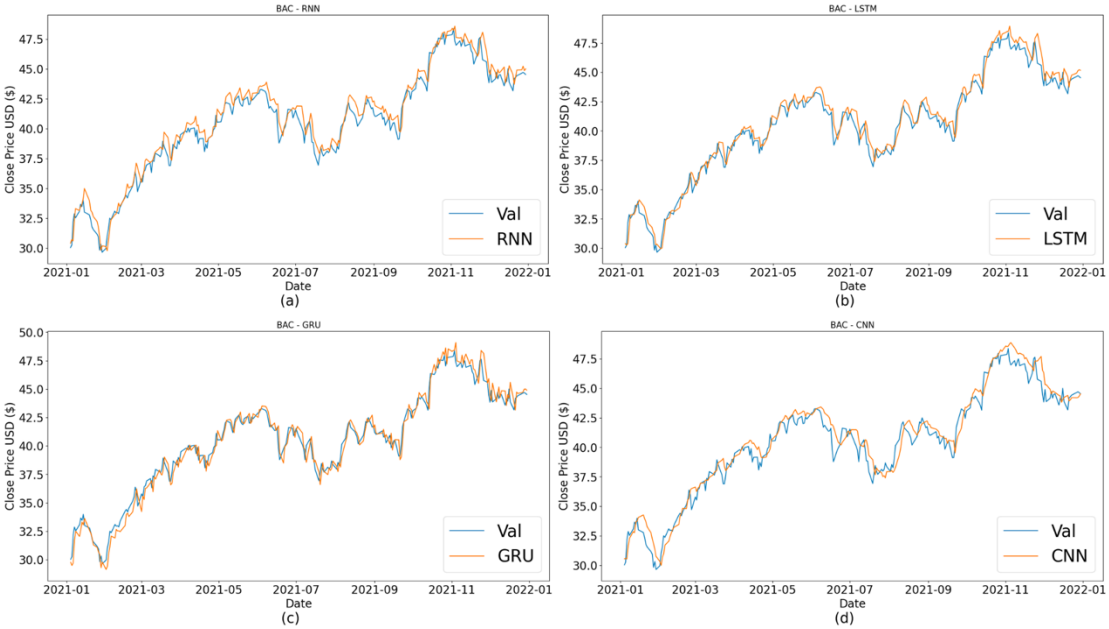

Figure 11: BAC prediction vs actual closing price using (a) VRNN; (b) LSTM; (c) GRU; (d) CNN

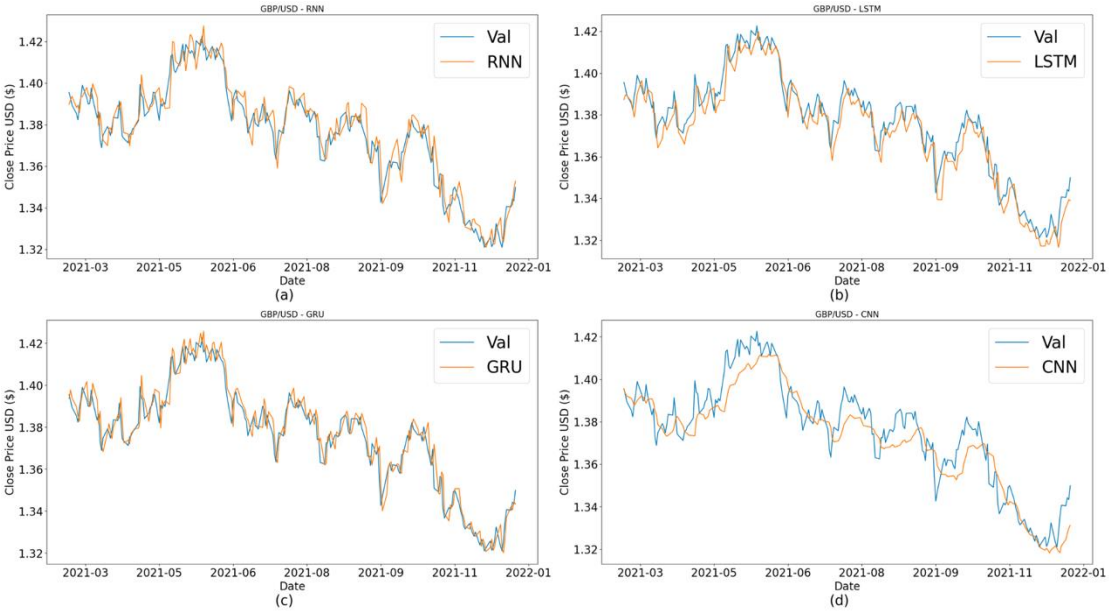

Figure 12: GBP/USD prediction vs actual closing price using (a) VRNN; (b) LSTM; (c) GRU; (d) CNN

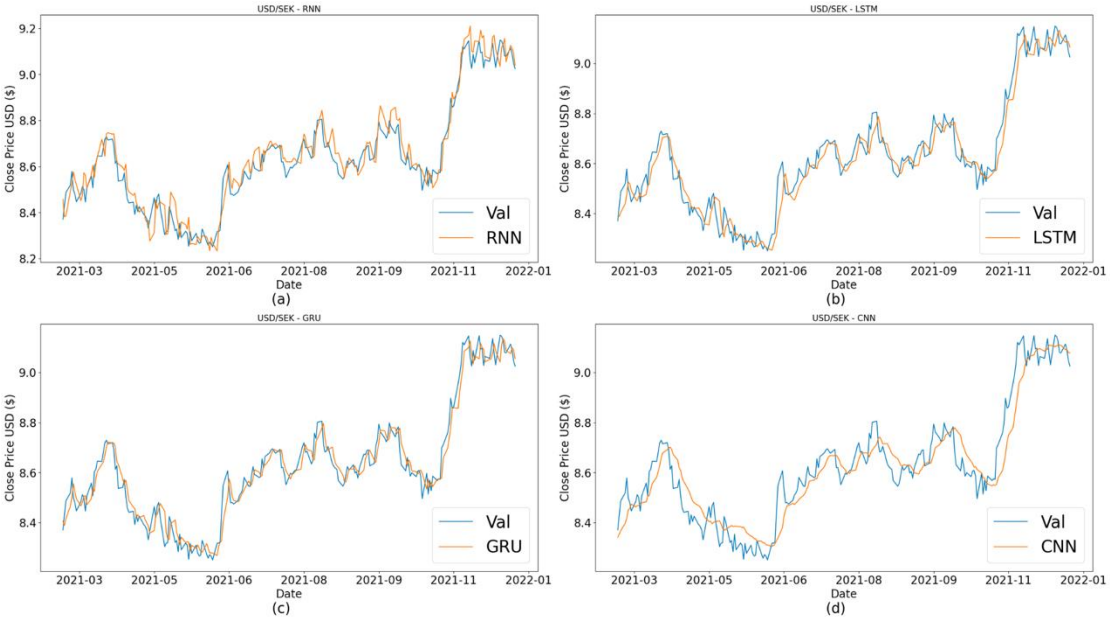

Figure 13: USD/SEK prediction vs actual closing price using (a) VRNN; (b) LSTM; (c) GRU; (d) CNN

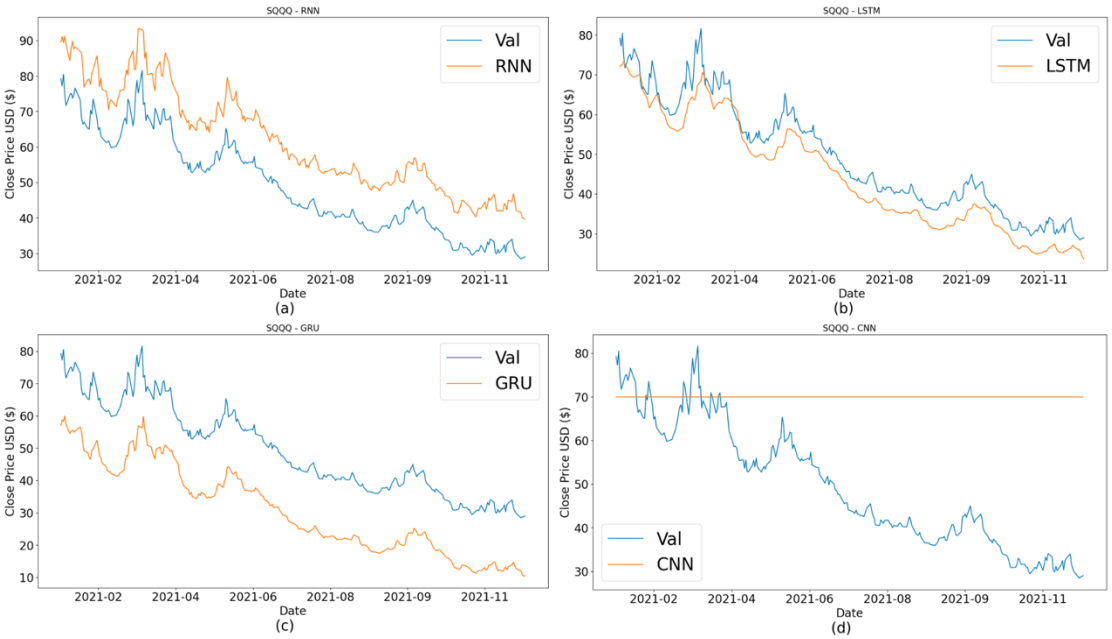

Figure 14: SQQQ prediction vs actual closing price using (a) VRNN; (b) LSTM; (c) GRU; (d) CNN

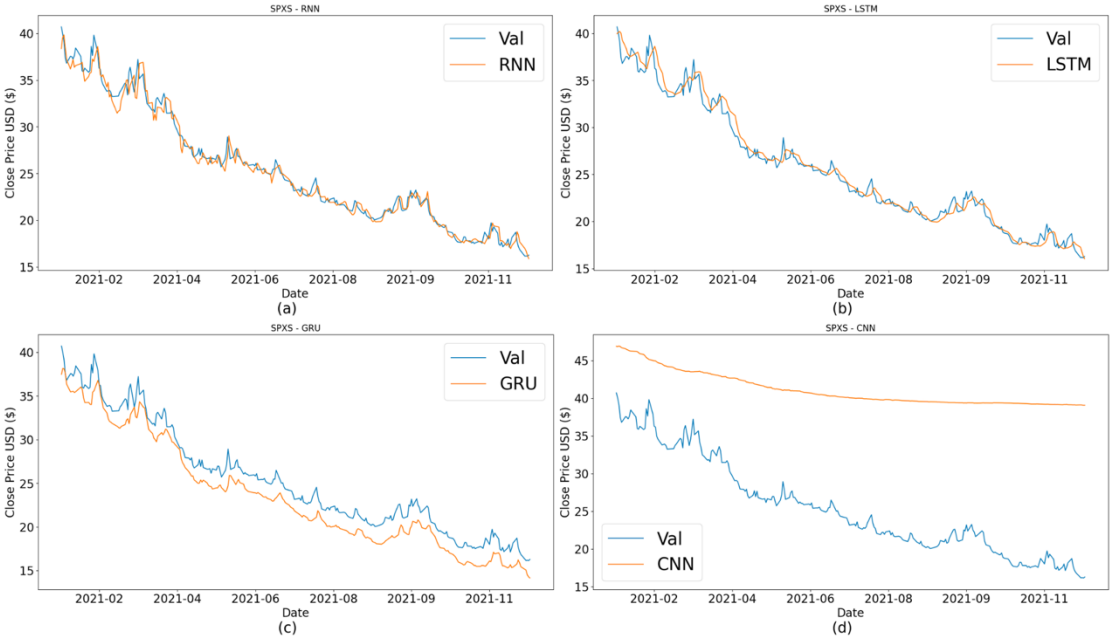

Figure 15: SPXS prediction vs actual closing price using (a) VRNN; (b) LSTM; (c) GRU; (d) CNN

For stocks and currencies, the LSTM, VRNN and GRU models have generally managed to capture the overall trend of the validation relatively well as seen throughout Figures 8 to 15, whereas the CNN models seem to have a delay in its prediction where it displays a peak in its predictions after the peaks already occurred in the validation. Amongst the variants of RNNs, the GRU model managed to capture the changes in the trend with the highest accuracy as compared to the other models.

For ETFs, the CNN model faces difficulty to capture the overall trend for both SQQQ and SPXS as seen in Figures 14 and 15 where the predictions plateaus and flatlines. In the case of SQQQ, the LSTM, VRNN and GRU models have managed to mimic the overall trend but did not fare well in terms of accuracy as in Figure 14 where the plotted predictions are distant from their validations with LSTM having the highest accuracy. In the case of SPXS, the LSTM model also has the highest accuracy for SPXS, beating the VRNN model by a little.

In terms of execution time from Table 2, the CNN models are clearly the fastest. This is because of how the model interprets the data fed to it. The CNN models interpret data as a grid like structure that resembles a matrix, whereas the other RNN variants reads the data in sequence which explains why it takes significantly longer to execute as compared to the CNN model.

7.2. *Discussion*

The purpose of retrieving data from different stock exchanges is to see how well the different deep learning models would work on different types of data.

For stocks and currencies, the VRNN model was outperformed by the GRU model as it cannot store sequences that are too long, causing it to only look at the latest closing prices available for forecasting which might lead to poorer accuracy in its forecast. The GRU model on the other hand, with its update and reset gates were able to decide which data is needed to pass along or neglected. As the stock and currency data are more volatile, the GRU model was able to outperform the LSTM model too probably because the forget gate present in it prematurely discarded some data points, making future predictions less accurate.

However, in the case of ETFs, the LSTM model surprisingly outperforms the GRU when forecasting ETFs. Looking at the historical data of SQQQ and SPXS, there is clearly a continuous downward trend of the share price. At instances like this, the forget gate in the LSTM model comes in handy as the past data gives no prediction value in forecasting. Hence, the model was able to make predictions with only data that is relevant for output.

From Table 1, it shows that the CNN model is outperformed by every other model for all cases. CNN model only accounts data that is in the specified window. Therefore, its inability to consider past data has caused its predictions to be premature or imprudent when dealing with long term predictions, as compared to the other variants of RNNs, which involves a recurring process of feeding outputs back into the model. CNN is more suitably applied in deep learning applications involved in image recognition as they are preferred in deciphering visual, sparse, and or random data that is "sequence-less".

The proposed CNN model might also be ill-equipped with the necessary tools for it to be on par with the other RNN variants which can be improved by further tuning it such as by increasing the number of epochs (Hiransha *et al*. 2018; Selvin *et al*. 2017). Increasing the number of epochs can benefit the CNN by allowing it to go through the data set a few more times before processing the forecast, this is because CNNs gather all the information and its parameters at the beginning, in contrast to RNNs whose parameters are constantly updated after each input is passed through it. Earlier, three epochs were set for all models for the sake of uniformity. This has proven to be insufficient for the CNN, causing it to be at a disadvantage. Setting the epochs at 75 with some fine tuning of the layers, the new RMSE and MAPE values of the CNN models have greatly improved as tabulated in Table 3 which can also be seen in Figure 16.

Table 3: Comparison of RMSE and MAPE between CNN with 3 epochs and 75 epochs

	RMSE		MAPE	
epoch	3	75	3	75
AAPL	5.488	2.857	0.033	0.016
TSLA	20.933	17.081	0.061	0.050
ROKU	30.626	27.071	0.073	0.061
BAC	0.926	0.773	0.018	0.015
GBP/USD	0.009	0.006	0.005	0.004
USD/SEK	0.076	0.063	0.007	0.006
SQQQ	24.951	20.479	0.549	0.277
SPXS	16.240	4.523	0.693	0.189

Another way to better the CNN's forecast accuracy is to include additional features like opening price and volume as CNN models are better suited for multivariate analysis (Chen & Huang 2021).

Overall, there is no one single model that fits universally into all types of historical financial market data as all four models gave decent forecasts on the test sets. CNNs, a neural network that typically handles image classification problems can also be useful in processing time series forecasts by incorporating an extra data preprocessing step to allow the model to process the time series as an image (Ozbayoglu *et al*. 2020). This opens the possibility of other types of deep learning models that could also be implemented to carry out time series analysis, so it would be ideal to test out different models when applying deep learning for time series forecasts. This is unfortunately motivated by the lack of understanding towards how deep

learning models would react to different types of data that exhibit varying qualities of volatility and dependencies. Khaldi *et al*. (2023) did a comprehensive investigation into which RNN architecture would best suit each time series behavior which calls for preliminary analysis on the historical stock price data to be considered when choosing the best model for different investments.

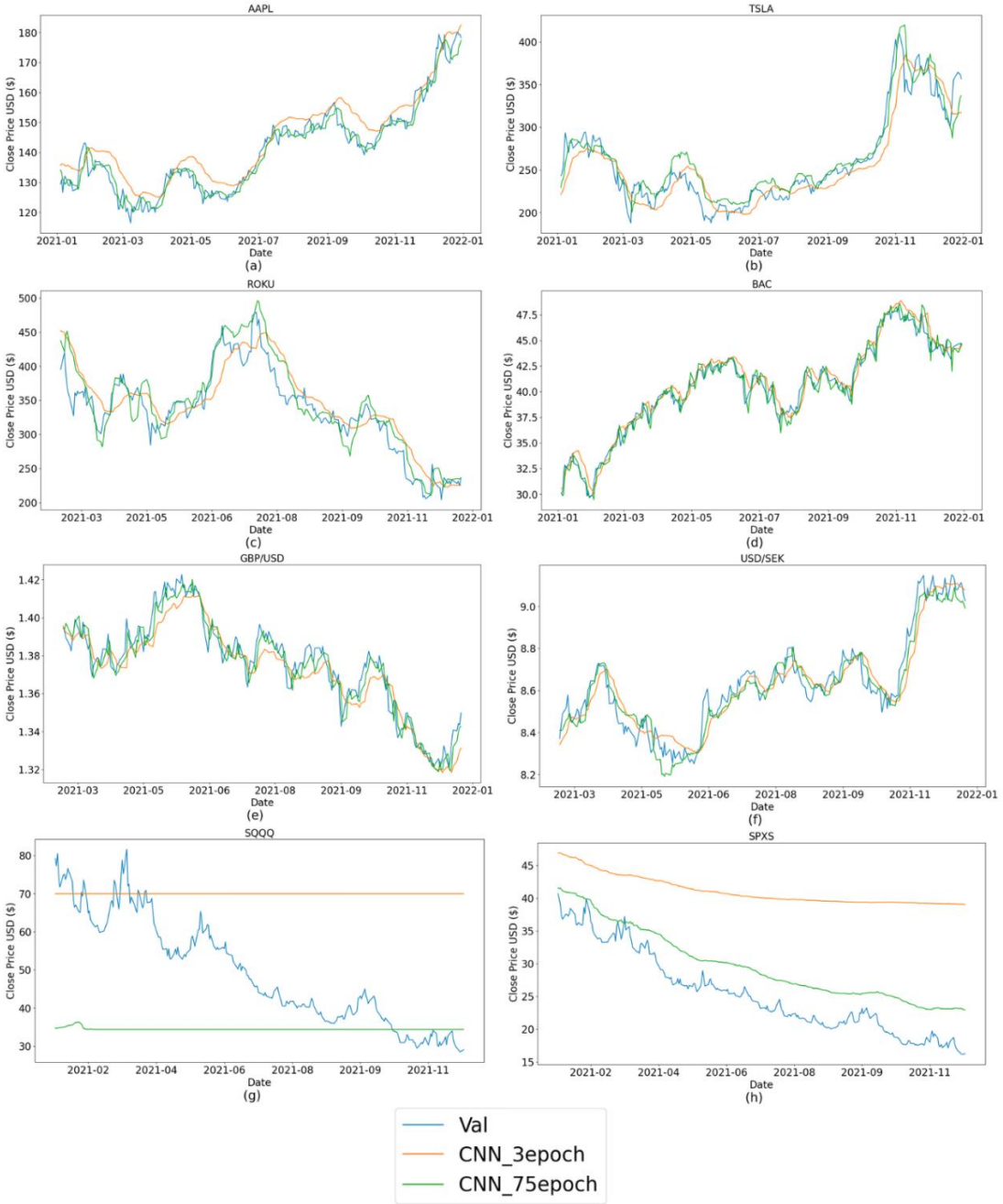

Figure 16: Comparison of predictions of CNN with 3 epochs and 75 epochs vs actual closing price for (a) AAPL; (b) TSLA; (c) ROKU; (d) BAC; (e) GBP/USD; (f) USD/SEK; (g) SQQQ; (h) SPXS

8. Conclusion

In this work, four different deep learning models (VRNN, LSTM, GRU and CNN) have been experimented on to gauge their ability of forecasting the closing price of various investments from the financial market. Results show that GRU outperforms all the other models when it comes to forecasting data that are more volatile, while LSTM is better suited for data that displays an overall trend and is less volatile. The VRNN falls short of GRU and LSTM due to the absence of gates. Nevertheless, the three variants of recurrent neural networks performed relatively well in capturing the dynamics of the data. CNN however is outperformed by the other models possibly due to its inability to forecast using data that falls outside of the time step window, as well as its requirement of better hyperparameter tuning like the number of epochs.

For future works, the models can be better improved by allowing multivariate analysis on the data using multiple features such as the trading volume and the opening price of the investment. More intuitive ways to determine the hyperparameter settings of both CNNs and RNNs should be researched on so that the full potential of the model can be accessed as the performance of the model greatly depends on it. Hybrid models can also be engineered to include layers of different deep learning models to combine the strengths of each model in synergy such as LSTM-GRU (Hossain *et al*. 2018) and CNN-GRU (Jaiswal & Singh 2022).

Acknowledgments

The authors would like to thank the reviewers for their insigts which led to the improvement of the paper.

References

Ariyo A.A., Adewumi A.O. & Ayo C.K. 2014. Stock price prediction using the ARIMA model. In *Proceedings of the 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, pp. 106-112.

Babu C.N. & Reddy B.E. 2015. Prediction of selected Indian stock using a partitioning–interpolation based ARIMA–GARCH model. *Applied Computing and Informatics* **11**(2): 130-143.

Chen Y.C. & Huang W.C. 2021. Constructing a stock-price forecast CNN-model with gold and crude oil indicators. *Applied Soft Computing* **112**: 107760.

Cho K., van Merriënboer B., Bahdanau D., & Bengio Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In. *Proceedings of the SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111.

Chollet F., & others. 2015. Keras. *Github*. Retrieved from https://github.com/fchollet/keras.

Di Persio L. & Honchar O. 2016. Artificial Neural Networks architectures for stock price prediction: Comparisons and applications. *International Journal of Circuits, Systems and Signal Processing* **10**: 403-413.

Dong Y.C., Li S.Y. & Gong X. 2017. Time Series Analysis: An application of ARIMA model in stock price forecasting. In *Proceedings of the 2017 International Conference on Innovations in Economic Management and Social Science (IEMSS 2017)*, pp. 703-710.

Granger C.W.J. & Newbold P. 1986. *Forecasting Economic Time Series*. 2nd Ed. Orlando, FL: Academic Press, Inc..

Hiransha M., Gopalakrishnan E.A., Vijay K.M. & Soman K.P. 2018. NSE stock market prediction using deep-learning models. *Procedia Computer Science* **132**: 1351-1362.

Hochreiter S. & Schmidhuber J. 1997. Long short-term memory. *Neural Computation* **9**(8): 1735–1780.

Hossain M.A., Karim R., Thulasiram R., Bruce N.D.B. & Wang Y. 2018. Hybrid deep learning model for stock price prediction. In *Proceeding of the 2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1837-1844.

Jain S., Gupta R. & Moghe A. 2018. Stock price prediction on daily stock data using deep neural networks. In *Proceedings of the 2018 International Conference on Advanced Computation and Telecommunication (ICACAT)*, pp. 1-13.

Jaiswal, R. & Singh, B. 2022. A hybrid convolutional recurrent (CNN-GRU) model for stock price prediction. In *Proceeding of the 2022 IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT)*, pp. 299-304.

Khaldi R., El Afia A., Chiheb R. & Tabik S. 2023. What is the best RNN-cell structure to forecast each time series behavior? *Expert Systems with Applications* **215**: 1199140.

Kingma D.P. & Ba J. 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980

Kumar R., Kumar P., & Kumar Y. 2021. Analysis of financial time series forecasting using deep learning model. In *Proceedings of the 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 877-881.

LeCun Y., Bottou L., Bengio Y. & Haffner P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **86**(11): 2278-2324.

Mohammad J.H. & Owda A.Y. 2021. A novel Cryptocurrency price prediction model using GRU, LSTM and bi-LSTM machine learning algorithms. *AI* **2**(4): 477-496.

Nielsen A. 2019. *Practical Time Series Analysis: Prediction with Statistics and Machine Learning*. 1st Ed. California: O'Reily Media, Inc.

Ozbayoglu A.M., Gudelek M.U. & Sezer O.B. 2020. Deep learning for financial applications: A survey. *Applied Soft Computing* **93**: 106384.

Preeti D.A., Bala R. & Singh R.P. 2018. Financial time series forecasting using deep learning network. In Deka G., Kaiwartya O., Vashisth P. & Rathee P. (eds.). *Communications in Computer and Information Science* **899**: 23-33. Singapore: Springer.

Sako K., Nyunga B.N. & Rodrigues P.C. 2022. Neural networks for financial time series forecasting. *Entropy* **24**(5): 657.

Saud A.S. & Shakya S. 2020. Analysis of look back period for stock price prediction with RNN variants: A case study on banking sector of NEPSE. *Procedia Computer Science* **167**: 788-798.

Selvin S., Vinayakumar R., Gopalakrishnan E.A., Menon V.K. & Soman K.P. 2017. Stock price prediction using LSTM, RNN and CNN-sliding window model. In *Proceedings of the 2017 International Conference on Advances in Computing, Communications, and Informatics (ICACCI)*, pp. 1643-1647.

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

Sparks J.J. & Yurova Y.V. 2006. Comparative performance of ARIMA and ARCH/GARCH models on time series of daily equity prices for large companies. University of Illinois, Chicago, USA.

Institute of Mathematical Sciences
Faculty of Science
Universiti Malaya
50603 Kuala Lumpur
E-mail: zhongjingyap@gmail.com, dharini@um.edu.my[*]

Received: 30 April 2023
Accepted: 25 May 2023

[*]Corresponding author