Hybrid Deep Learning Model for Stock Price Prediction

Mohammad Asiful Hossain
Department of Computer Science
University of Manitoba
Winnipeg, Canada
hossaima@cs.umanitoba.ca

Rezaul Karim
Department of Computer Science
University of Manitoba
Winnipeg, Canada
karimr@cs.umanitoba.ca

Ruppa Thulasiram
Department of Computer Science
University of Manitoba
Winnipeg, Canada
tulsi@cs.umanitoba.ca

Neil D. B. Bruce
Department of Computer Science
University of Manitoba
Winnipeg, Canada
bruce@cs.umanitoba.ca

Yang Wang
Department of Computer Science
University of Manitoba
Winnipeg, Canada
ywang@cs.umanitoba.ca

***Abstract*—In this paper, we propose a novel stock price prediction model based on *deep learning*. With the success of deep learning algorithms in the field of Artificial Neural Network (ANN), we choose to solve the regression based problems (stock price prediction in our case). Stock price prediction is a challenging problem due to its random movement. This hybrid model is a combination of two well-known networks, Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). We choose the S&P 500 historical time series data and use significant evaluation metrics such as mean squared error, mean absolute percentage error etc., that conventional approaches have used. In experiment section, we have described the effectiveness of each of the component of our model along with its performance gain over the state-of-the-art approach. Our prediction model provides less error by considering this random nature (change) for a large scale of data.**

***Index Terms*—LSTM, GRU, DNN, Hybrid-network, Stock price prediction**

I. Introduction

In recent years, stock price prediction has been one of the continuously studied area in computational finance research. Stock price prediction is one of the most challenging task due to its highly dynamic nature. The movement of price seems to be highly random event and capturing price movement is a challenging task. With the success of Artificial Neural Network (ANN) in different prediction tasks, researchers are more interested to deploy high level machine learning approaches to build better prediction models. Some of the modern application areas of ANN include autonomic car driving, robot aided surgery, guidance of intelligent drones, software development, nuclear physics, and finance [1].

Deep Neural Network (DNN) is an advanced branch of ANN. During past few decades, DNN has gained enormous popularity due to its effectiveness on prediction based problems. DNN has been showing promising performance for problems such as speech recognition, natural language processing, computer vision, robotics, computational finance. By considering these as motivation, we propose a stock price prediction model based on DNN. Stock price prediction is a regression based problem. The problem of predicting stock price can be formally stated as, given historical data (open/ close/ high/ low price, trading volume etc.) of last n number of days for a particular stock or index, predict the next day closing price.

In applying advanced machine learning approach for stock price prediction, the problem is formulated as a regression based problem. With the evolution of Recurrent Neural Network (RNN) [2] in different research areas, researchers started applying RNN to solve regression based problems with sequence data. One most important problem encountered while training the network using RNN is that it can not keep track of long input dependencies [3] while training. To process big dataset, RNN and conventional approaches take long time. The main goal of this research is to provide a faster and more accurate neural network method.

To achieve our goal, we propose hybrid model that consists of two well-known DNN approaches: Long Short Term Memory (LSTM) [4] and Gated Recurrent Unit (GRU) [5]. These architectures allow us to deal with providing prediction based on (historical) stock prices. We have done extensive experiments to show the effectiveness of the proposed hybrid model in the experiment section. The architecture of propose network is simple enough to deal with training, testing and validating the dataset. Our dataset is S&P 500 time series data. The major contributions of this study are:

- Our proposed model is trained on large historical data (about 66 years). We compare our model with state-of-the-art work in literature and show that our model yields better result in terms of mean square error(MSE).
- The proposed model is capable of handling the large variations in stock price by avoiding over-fitting.
- To the best of our knowledge, this is a first attempt in using deep learning based hybrid approach to solve stock

978-1-5386-9276-9/18/$31.00

price prediction. More deep learnign research int he field of computational finance may follow this model.

Rest of this manuscript is organized as follows: In section II we present published literature related to price prediction in general, and models using ANN and DNN in particular. We present our proposed model in section III. We describe our experimental framework and results in section IV followed by conclusion and future work section V.

II. Related Work

There have been notable research efforts to predict future value or movement of stock prices using technical analysis of historical data like opening price, closing price, trading volume etc. Most of the research formulate the problem of the future prediction in either of the two categories, future movement prediction or future value prediction. First one is formulated as classification problem and the other as regression problem. A great deal of research have been focused to apply machine learning and neural network approaches to address the problem of prediction motivated by recent success of neural networks in various other fields including natural language processing and computer vision. In this section, we present some most recent artificial neural network based research on stock price prediction that are closely related to our work.

Guresen et al. [6] have analyzed and evaluated the effectiveness of different neural network models including multilayer perceptron (MLP) [7], dynamic artificial neural network (DAN2) [8] and the hybrid neural networks [9], which use generalized autoregressive conditional heteroscedasticity (GARCH) model to extract new input variables. They have reported comparison for each model with MSE and MAD using real exchange daily rate values of NASDAQ Stock Exchange index. Adebiyi et al. [10] have presented MLP neural networks applied to predict the future stock price and showed very good prediction on different stock symbols. They have experimented with MLP neural networks with one hidden layer and sigmoid activation function. Previous days' open, close, high and low prices, trading volumes and some fundamental analysis variables have been used as input to the network.

Recently, Di Persio et al. [1] have compared MLP, convolution neural network (CNN) and recurrent neural network (RNN) performance on predicting both stock price movement and future price. They have used S&P500 index data from 1950 to 2016 for their experiment with different types of neural networks. For price prediction, they have used previous days open/close/high/low data as input and MSE and accuracy as performance measure. They have also presented wavelet-CNN [11], [12] and ensemble methods for price prediction. In the presented results, it seems all of the neural networks methods are very competitive and in their hyper-parameter configurations, CNN have had least MSE. The most interesting part of their research is that they have worked with more deeper networks compared to previous works. For the RNN, they have used LSTM network.

RNN with Gated Recurrent Units (GRU) have been applied for stock price movement prediction in some recent works, for example, [13] and have shown promising success. In [13], Bidirectional Gated Recurrent Unit (BGRU) have been used to predict stock price movements, and their model achieved accuracy of nearly 60% in S&P500 index prediction and the individual stock prediction is over 65%. Hence, the potential of recurrent neural networks with gated recurrent units or combination of LSTM and GRU needs to be studied in depth for stock price prediction too.

From the motivation of success of deep RNN with LSTM and GRU units in different areas to model sequence data, in this study, we attempt to experiment with different types of RNN with LSTM and GRUs to predict future stock price based on previous days data. IEEEtranS

III. Hybrid Network

In this section, we provide the details of the proposed hybrid network. From Figure 3, it is clear that the hybrid model is basically a combination of Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks. Both of the LSTM and GRU are powerful recurrent network that can perform better and faster in terms of accuracy in regression based prediction. Stock price prediction is a regression based problem, the main intuition behind using these networks. Firstly, we pass the input to the LSTM network which will generate first level prediction. Secondly, we pass the output of LSTM layer to the GRU layer to get the final prediction. Our full network architecture is as follows: LSTM layer, Dropout Layer, GRU Layer, Dropout Layer, and a Dense Layer. We demonstrate the functionality of both of the network in later sub-sections.

A. LSTM Unit

LSTM [4] is a derivative of Recurrent Neural Network (RNN), which operates on sequential data. RNN has been proved as one of the best methods used to solve some popular problems like speech-recognition, test pattern recognition, text generation etc. To solve the long term dependencies [3] problem, Hochreiter et al. [4] proposed LSTM which has a memory unit that can keep track of the certain amount of training data. One more advantage of using LSTM is that it can learn automatically when to forget the memory for a particular sequence. The LSTM transition equations are given as follows [13]:

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

$$f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}) \quad (2)$$

$$o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}) \quad (3)$$

$$u_t = \tanh(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}) \quad (4)$$

$$c_t = i_t \odot u_t + f_t \odot c_{t-1} \quad (5)$$

$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

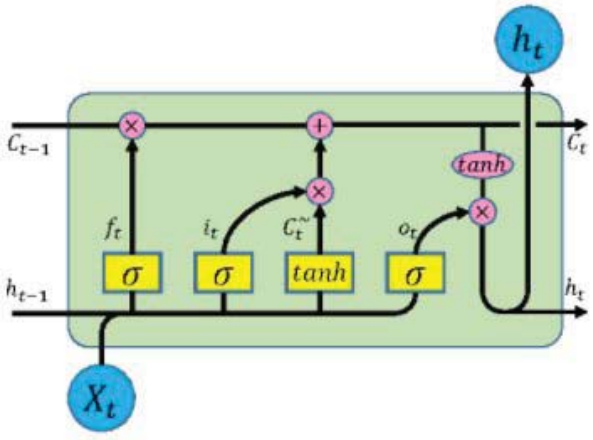

Fig. 1. LSTM unit [14]

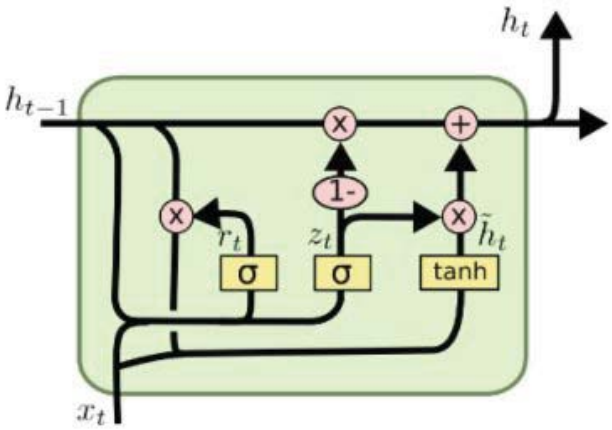

Fig. 2. GRU [15]

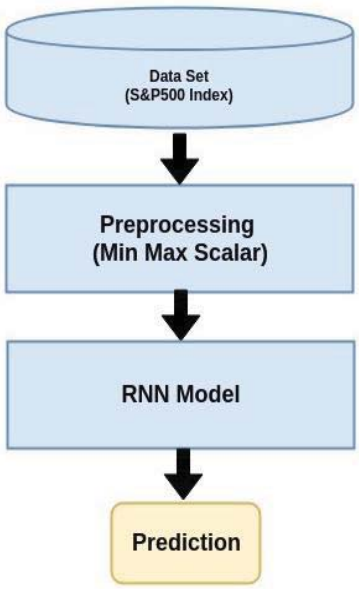

Fig. 3. Proposed Network

From the above equations, for an input vector the LSTM unit at time step t: i_t is an input gate, f_t is a forget gate, o_t is an output gate, c_t is a memory cell, u_t is an activation function and hidden state h_t. The default connections among these units are presented in Figure 1.

B. GRU

Another recent version of RNN is known as Gated Recurrent Unit proposed by Cho et al. [5]. The main difference between LSTM and GRU is, GRU combines the forget and the input gates into a single update gate. It also merges the cell state and the hidden state and makes some other changes [13]. GRU model is simpler yet faster network than the standard LSTM models although the basic purpose of using GRU is similar as LSTM. The internal mechanics of the GRU are defined as [13]:

$$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1} + b^{(z)}) \tag{7}$$

$$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{t-1} + b^{(r)}) \tag{8}$$

$$a_t = \tanh(Wx_t + r_t \odot Uh_{t-1} + b^{(h)}) \tag{9}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot a_t \tag{10}$$

From the above equations, z_t is update gate, r_t is reset gate, a_t is activation function, h_t is hidden state output gate. Figure 2, represents the configuration of GRU network.

C. Hybrid Network

The proposed model in Figure 3, is a combination of LSTM and GRU. First, we pass the input vector to the LSTM unit with one hidden layer and we get the output F_{lstm}. Next, we pass F_{lstm} to as the input of GRU unit. At the second layer we get the F_{gru} as the output of GRU unit. Then, we pass F_{gru} into dense network followed by a linear activation. Finally, we calculate Euclidean loss (L2 loss), between the prediction and the ground truth and back propagate this loss to the network to update the model parameters. After 20 epochs by demonstrating loss curve we get our trained model. In section IV, we analyze the effectiveness of our proposed method.

D. Loss Function

As the stock price changes randomly, we train our network based on mean square error loss. We calculate the euclidean distance between the actual value and prediction for a particular symbol class. The loss function is given as:

$$L(\Theta) = \frac{1}{N}\sum_{i=1}^{N}(F(X_i;\Theta) - Y_i)^2 \tag{11}$$

where, Θ is a set of learnable parameters in proposed method. N is the vector containing training data. X_i is the i-th input

and Y_i is the actual value for X_i. $F(X_i;\Theta)$ stands for the predicted value generated by the proposed method which is parameterized with Θ for sample X_i. $L(\Theta)$ is the loss between prediction and the actual value.

IV. Experimental Result

A. Implementation Detail

Our network is implemented in python platform using Keras library [16]. LSTM and GRU networks are by default implemented in Keras, which help us a lot to customize the network. Moreover, we use Adam optimizer [17], learning rate is set to 0.001. 20 epochs are used to train the network in Intel core i7 processor with processing speed 4GHz and 32GB RAM (Random Access Memory). Implementation of the full network in detail can be viewed from Fig 4.

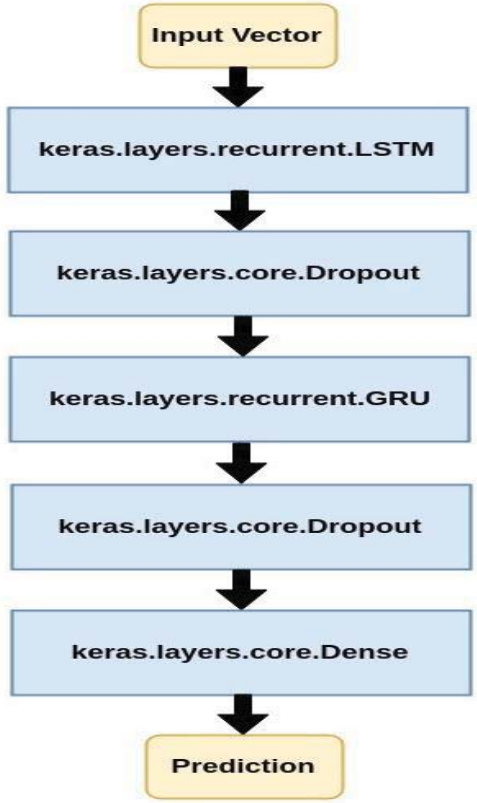

Fig. 4. Hybrid Network Architecture

B. Evaluation Metrics

We have used mean absolute error (**MAE**), mean squared error (**MSE**) and mean absolute percentage error (**MAPE**) metrics for evaluation as we have formulated our problem as a regression problem. Equations representing these metrics are given below:

$$MAE = \frac{1}{N}\sum_{t=1}^{n}|y_t - \hat{y_t}| \tag{12}$$

$$MSE = \frac{1}{N}\sum_{n=1}^{N}(y_t - \hat{y_t})^2 \tag{13}$$

$$MAPE = \frac{100\%}{N}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y_t}}{y_t}\right| \tag{14}$$

Where y_t is the actual value and $\hat{y_t}$ is the predicted value for t^{th} day.

Method	MSE	MAE	MAPE (%)
MLP [1]	0.26	–	–
CNN [1]	0.2491	–	–
RNN [1]	0.2498	–	–
Average Ensemble [1]	0.23	–	–
Hand-weighted Ensemble [1]	0.23	–	–
Blended Ensemble [1]	0.226	–	–
Proposed Method	**0.00098**	0.023	4.13

TABLE I
Results on S&P500 historical data compared with previous methods on test data

C. Dataset

For regression based problems like stock price prediction, data processing is an important issue which has direct impact on over all performance. We have downloaded S&P500 index historical data of 66 years from 1950 to 2016, using *Yahoo Finance* API. The columns of downloaded raw data can be view as ['Date','Open','Close','Volume']. We split the dataset into training , testing and validation set. For training set we take the first 80% of the data and for testing we choose rest of the 20% data. The training set contains historic data from 1950 to approximately the third quarter of 2002, and the rest of the data till 2016 are in test set. 10% of the training data have been used for validation as well. We have used MinMaxScaler [18] for preprocessing the data and changed the range from 0 to 1. This preprocessing preserves the relative patterns but changes only the range of values by scaling. The intuition behind preprocessing the dataset is to make it compatible for proposed hybrid network. For fair comparison, we follow the same setting as described by Persion and Honchar [1]. In Figure 5, we present the normalized training data form 1950 to 2002. The curve shows the randomness of stock price data. Our main goal is to predict the closing price of next day. We put all the training data into a one dimensional vector and pass it to the network with a fixed learning rate(.001). Figure 6 shows the normalized test dataset representation, which is also showing the randomness of stock price.

D. Results

While training our full network we saved the losses for each input. In Fig 7 we draw a loss curve for 20 epochs for the proposed hybrid network (1 epoch is full iteration over all the training data). We can see that, our loss curve is going

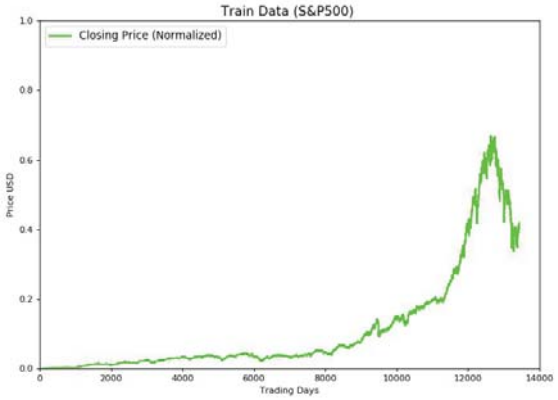

Fig. 5. Normalized Actual Closing Price (Train Data)

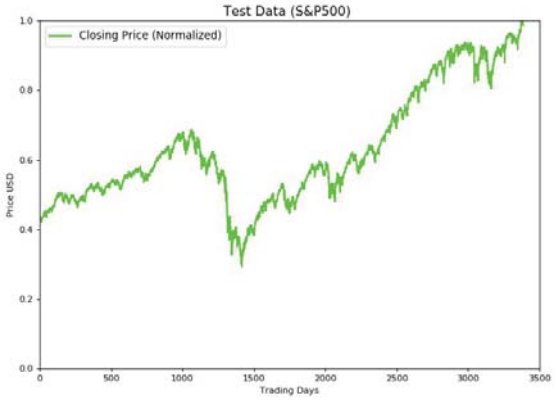

Fig. 6. Normalized Actual Closing Price (Test Data)

downward with the increase in epochs. From the design of our experiment setting 20 epochs are sufficient enough to find out the best model snapshot. In Table I, we present the comparative result. We can see that, proposed method outperforms all the methods mentioned in baseline work [1]. We achieved MSE of 0.003 and MAPE 4.13% which is a significant improvement for stock price prediction problem.

E. Ablation study

In this section, we describe the effect of every component of our propose network. To start with, first we train our network with just 1 layer LSTM, and during evaluation we get .027 MAE. Then, we evaluate 1 layer GRU to see its performance. From table II we can see that 1 layer LSTM is performing better than 1 layer GRU in terms of all of the metrics MAE, MSE and MAPE. Interestingly, from Figure 8 we observe that the single layer LSTM tends to overprice or predict higher than actual value most of the time and from Figure 11, we observe that single layer GRU often tends to under-price, mostly at the time of rapid movement. This observation inspires us

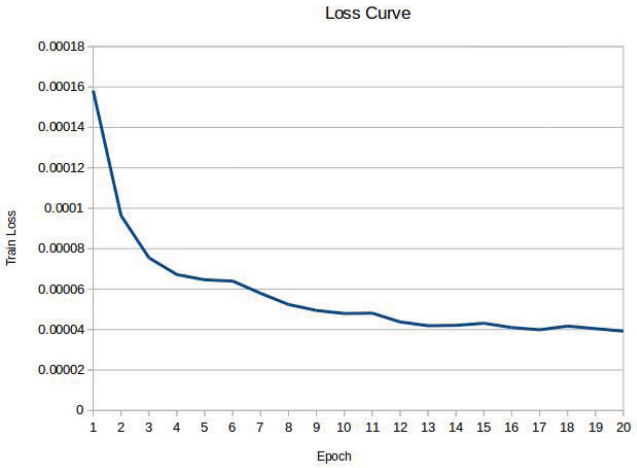

Fig. 7. Training Loss

of combining these two networks (LSTM, GRU) to get an optimized prediction. We have also gone deeper with only LSTM and GRU. From table II we observe that going deeper with LSTM network, for example, 2 and 3 layers (Figures 9 and 10) does not improve the prediction quality based on the used metrics. For the GRU network, going deeper up to 2 layer (Figure 12) gives some improvement over single layer GRU network but degrades again at 3 layer GRU network. Also, we can see that 2 layer GRU (Figure 12) performance is close to 1 layer LSTM (Figure 8).

By considering this, we propose our hybrid network by combining both LSTM and GRU to offset the underpricing and overpricing of each other and get better prediction. For this we have added a GRU layer following the LSTM layer. We also evaluate the alternative hybrid setting by putting GRU as first layer and LSTM as second layer. From Table II we can see that our proposed hybrid setting outperforms the alternative hybrid setting. Table II demonstrates that 1 layer LSTM results close to our proposed model where 3 layer architecture is not giving improvement anymore. It proves that if we add more layers or make the network deeper with same components it might cause overfitting and degrade the performance. The hybrid network have been found to be best performing in terms of the metrics used as presented in table II. As shown in Figure 14, the prediction of the proposed hybrid model is very close to actual values compared to other network configurations.

Closeness Analysis: To demonstrate the intuition behind proposing this hybrid network we have some analysis by plotting some graphs regarding different networks. On these graphs we mainly plot the prediction and actual value line for a certain duration. From Figures 11 and 8, we can see that both of the network is doing over pricing. Prediction line is above the actual value line. In Figures 12 and 9, these are the plots of 2 layer networks. For these networks, we can see that sometimes prediction is overestimating and sometimes its underestimating. For three layers in Figures 13 and 10], we

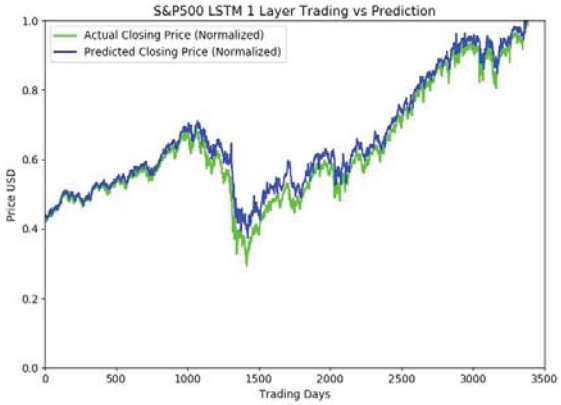

Fig. 8. SP500 Prediction vs Actual Closing Price (Single layer LSTM)

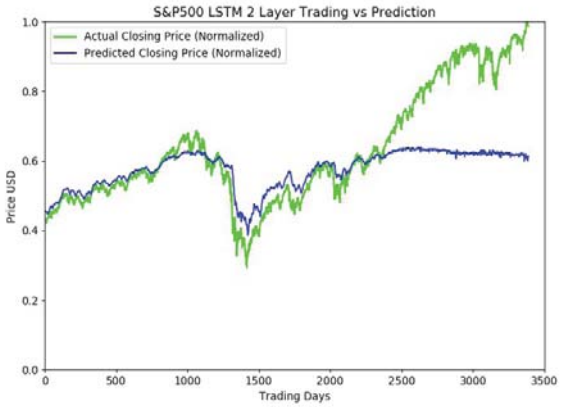

Fig. 9. SP500 Prediction vs Actual Closing Price (Two layer LSTM)

Fig. 10. SP500 Prediction vs Actual Closing Price (Three layer LSTM)

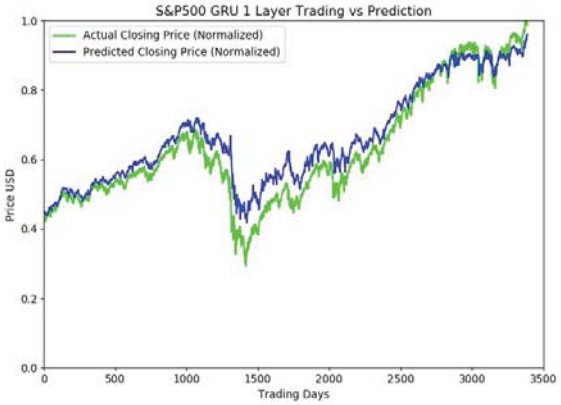

Fig. 11. SP500 Prediction vs Actual Closing Price (Single layer GRU)

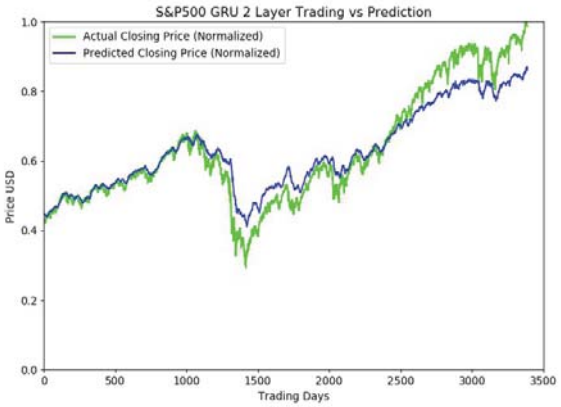

Fig. 12. SP500 Prediction vs Actual Closing Price (Two layer GRU)

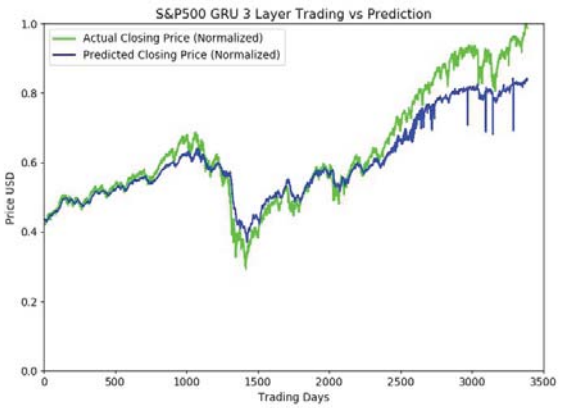

Fig. 13. SP500 Prediction vs Actual Closing Price (Three layer GRU)

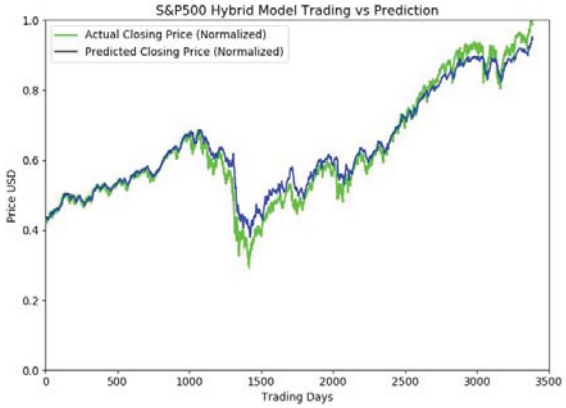

Fig. 14. SP500 Prediction vs Actual Closing Price (Proposed hybrid network)

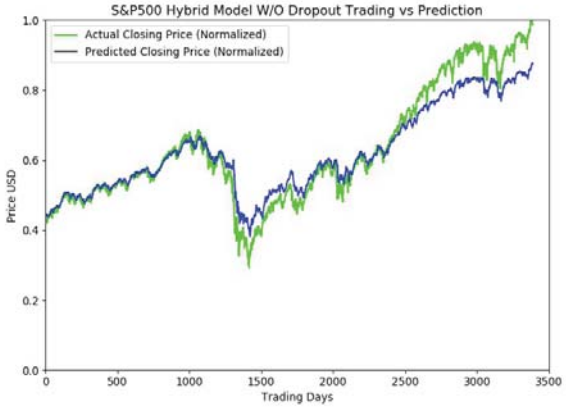

Fig. 15. SP500 Prediction vs Actual Closing Price (Proposed hybrid network w/o dropout)

Method	MAE	MSE	MAPE
LSTM (Single Layer)	0.027	0.001	4.918
GRU (Single Layer)	0.040	0.003	7.503
LSTM (Two Layers)	0.086	0.018	11.583
GRU (Two Layers)	0.028	0.001	4.649
LSTM (Three Layers)	0.051	0.003	9.173
GRU (Three Layers)	0.041	0.003	5.896
GRU+LSTM (dropout)	0.063	0.008	9.36
Proposed Model(w/o dropout)	0.031	0.002	4.667
Proposed Model	**0.023**	**0.00098**	**4.13**

TABLE II
EVALUATION ON TEST SET FOR DIFFERENT ARCHITECTURES

still see the same thing happening. To achieve the proper trade-off between overestimation and underestimation, our proposed method is showing promising result in Figure 14.

Overfitting Analysis: Overfitting a prime and common modeling error that might occur during training a neural network. It happens when a function is too closely fit to a limited set of data points. To handle this special type of error we used dropout layer after each of our network component. Dropout is mainly a regularization technique to make more training sample by deforming the given one during training [19]. With the help of this concept we figured out that our network performs better with dropout layer than the propose network without dropout layer. From Table II we can see that propose network with dropout is producing MSE 0.023 which better than proposed model without dropout which is producing MSE 0.031. The predictions for hybrid model without dropout is visualized in Figure 15 which when compared to predictions of proposed model in Figure 14 clearly identifies the importance of dropout layers used in the proposed model.

V. CONCLUSION

In this study, we have presented a deep recurrent neural network based stock price prediction model. We experimented with different recurrent neural network units including LSTM and GRU before proposing our deep recurrent neural network mode. Our final model is a network composed of both LSTM and GRU units, which is essentially an hybrid model. We have used dropout units with dropout probability 0.1 after both LSTM and GRU units. Final prediction layer of the models is a densely connected layer. We have presented experimental evaluation with S&P 500 historical data from 1950 to 2016. Our proposed hybrid network has achieved 0.00098 MSE in prediction with this dataset, which outperforms all the previous neural network approaches in the same dataset with very high margin.

As a future work, we intend to perform experiments with individual stock symbols and other financial predictions like foreign currency exchange rates. We expect that our findings will have a good contribution on the area of neural network based financial prediction model research.

ACKNOWLEDGMENT

The last three authors acknowledge Natural Sciences and Engineering Research Council (NSERC) Canada for partial financial support for this research through Discovery Grants. The first author acknowledge the International Graduate Student Entrance Scholarship from the Faculty of Graduate Studies, University of Manitoba and the Manitoba Graduate Scholarships(MGS). The second author acknowledge the International Graduate Student Entrance Scholarship from the Faculty of Graduate Studies, University of Manitoba. and Computer Science Entrance Awards.

REFERENCES

[1] L. Di Persio and O. Honchar, "Artificial neural networks architectures for stock price prediction: Comparisons and applications," *International Journal of Circuits, Systems and Signal Processing*, vol. 10, pp. 403–413, 2016.

[2] L. Medsker and L. Jain, "Recurrent neural networks," *Design and Applications*, vol. 5, 2001.

[3] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

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

[5] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
[6] E. Guresen, G. Kayakutlu, and T. U. Daim, "Using artificial neural network models in stock market index prediction," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10 389–10 397, 2011.
[7] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter, "The multilayer perceptron as an approximation to a bayes optimal discriminant function," *IEEE Transactions on Neural Networks*, vol. 1, no. 4, pp. 296–298, 1990.
[8] M. Ghiassi, H. Saidane, and D. Zimbra, "A dynamic artificial neural network model for forecasting time series events," *International Journal of Forecasting*, vol. 21, no. 2, pp. 341–362, 2005.
[9] L. Medsker, "Design and development of hybrid neural network and expert systems," in *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, vol. 3. IEEE, 1994, pp. 1470–1474.
[10] A. Adebiyi, C. Ayo, M. O. Adebiyi, and S. Otokiti, "Stock price prediction using neural network with hybridized market indicators," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 3, no. 1, pp. 1–9, 2012.
[11] S. Fujieda, K. Takayama, and T. Hachisuka, "Wavelet convolutional neural networks," *arXiv preprint arXiv:1805.08620*, 2018.
[12] Y.-H. Baek, O.-S. Byun, and S.-R. Moon, "Image edge detection using adaptive morphology meyer wavelet-cnn," in *Neural Networks, 2003. Proceedings of the International Joint Conference on*, vol. 2. IEEE, 2003, pp. 1219–1222.
[13] H. D. Huynh, L. M. Dang, and D. Duong, "A new model for stock price movements prediction using deep neural network," in *Proceedings of the Eighth International Symposium on Information and Communication Technology*. ACM, 2017, pp. 57–62.
[14] http://www.stratio.com/blog/deep-learning-3-recurrent-neural-networks-lstm/.
[15] https://pythonmachinelearning.pro/advanced-recurrent-neural-networks/.
[16] F. Chollet *et al.*, "Keras: Deep learning library for theano and tensorflow," *URL: https://keras. io/k*, vol. 7, p. 8, 2015.
[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[18] L. Al Shalabi, Z. Shaaban, and B. Kasasbeh, "Data mining: A preprocessing engine," *Journal of Computer Science*, vol. 2, no. 9, pp. 735–739, 2006.
[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.