ISSN 2278-3091
Volume 9, No.4, July – August 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse354942020.pdf
https://doi.org/10.30534/ijatcse/2020/354942020

Stock Prices Time Series Forecasting by Deep Learning using Three-Point Moving Gradient

Wong Pee Shoong[1], Siti Azirah Asmai[2], Tay Choo Chuan[3]
[1] Center for Advanced Computing Technology (C-ACT), Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia Melaka (UTeM), Melaka, MALAYSIA, Malaysia, p031810008@student.utem.edu.my,
[2]Center for Advanced Computing Technology (C-ACT), Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia Melaka (UTeM), Melaka, MALAYSIA, Malaysia, azirah@utem.edu.my.
[3]Faculty of Electrical Engineering, Universiti Teknikal Malaysia Melaka (UTeM), Melaka, MALAYSIA, Malaysia, tay@utem.edu.my

ABSTRACT

Everyone wants to know the future. For the knowledge of the future bring immeasurable opportunities, power and wealth. Anyone who can foresee what the future economic trends are, can generate enormous wealth. Thus, forecasting the stock market has always been a fascination among investors and speculators. Billions of dollars and tons of equipment have been poured into researching the movement of stock prices, with the hope of discovering the ultimate formula to strike gold in this venture. Forecasters employ various methods and apply various theorems to crack this elusive undertaking, but none has really succeeded accurately producing forecasts with acceptable margins of errors. One promising AI method is the Long Short-Term Memory (LSTM) method. Though LSTM is good at pattern recognition, it does not give much consideration to the strong affinity that binds the preceding and the succeeding values in a time series. It does not consider the relationship between the preceding and the succeeding values. This study proposes an alternative Artificial Intelligence-infused Long Short-Term Memory method (LSTM) which considers the preceding, present and succeeding values by forecasting the Three-point Gradient of the stock prices. By using reversed Linear Regression and the mean price, the precise forecast price can be reversed calculated with acceptable accuracies. The results may be just as accurate as normal LSTM method or even more if correctly tweaked and may require less process time

Key words: Gradient, Long Short Term Memory, Linear Regression, Stock Prices Forecasting

1. INTRODUCTION

The use of Long Short-Term Memory (LSTM) in stock price forecasting has been done for quite some time and it has achieved some remarkable results. LSTM is programmed to train the computer to best-fit the previous data in order to forecast the next value.

Stock price time series forecasting is one of the important topics in quantitative finance since accurate prediction of stock price is crucial [1]. It is crucial because the knowledge of future trends would assist investors in gaining optimal returns for their current investments. Kramer defines stock price as the company's market value divided by the number of shares it has issued. Kramer further states that the stock price reflects the value of a company's and this price only shows price changes in market capitalisation at a certain point in time [2]. Changes in the price will mirror changes in the company's worth. Market forces direct stock prices and they change daily due to supply and demand. The price would increase if the number of buyers (demand) outstrips the number of sellers (supply). Conversely, the price would fall if there are fewer buying activities than selling. In a nutshell, when more want it, it would become more expensive and when fewer demand of it, it would become cheaper. The venue for this purpose of transaction is the stock market. Corporate Finance Institute defines the stock market as public markets for issuing, buying and selling stocks that trade on a stock exchange. The role of the stock market is imperative to a country's economy as it is the fastest and most efficient way for companies to gather more capital. Through the stock market, the gathering of capital can be done in a swift, legal, orderly and secured manner.

Stock prices are not totally random. Stock prices do not fluctuate wildly. They exhibit some directed uptrends and downtrends. They have maximums and minimums. They have momentums. Hence, they are perceived to exhibit some obscure patterns and seasonal trends. Thus, forecasting stock prices to a fair amount of accuracy, is deemed logically possible if one can decipher their patterns and trend directions. The art of forecasting stock prices falls under the category of Time Series Forecasting. Much effort has been devoted over the past several decades to the development and improvement of Time Series Forecasting methods with the evolvement of various statistical and non-statistical approaches. Yet, the art of stock market forecasting remains elusive with inconclusive results. Pandey denotes that stock market forecasting is a technique that uses historical data as

inputs to make informed estimates that are predictive in determining the direction of future trends and having prior knowledge of any event, can help a company tremendously in the formulation of its goals, policies and planning [3]. Consequently, forecasting helps in determining a company's strategic business initiatives. There are many methods employed in stock market time series forecasting. Some are statistical while others are not. Brockwell and Davis listed among methods used in Time Series forecasting are Moving Average (MA), Autoregression Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA) [4]. Torres listed additional methods - Linear Regression, Exponential Smoothing, Dynamic Linear and Neural Network [5]. Shah, Isah and Zulkernine F (2019) conceded that there are many challenges in stock market forecasting [6]. In 2012, Brock highlights that the main problem with forecasting is determining what is to be forecasted – what are the forecasts that are beneficial to and can be fully utilised by the organization [7]. On the other hand, Rackauckas opines that ARMA and ARIMA are only suited for forecasting linear update models while LSTMs and RNNs are essentially for non-linear models [8]. Muller claims that LSTMs/RNNs are faster than traditional methods [9]. Calomme notes that LSTMs do not need a certain fixed input size thus making it a more flexible and adaptable method [10]. Cabrera et. al. (2019) stated that LSTMs solve long complex time lag tasks that never been solved by others [11]. Gers, Schmidhuber and Cummins also notices that "LSTM augmented by 'peephole connections' from its internal cells to its multiplicative gates, can learn the fine distinction between sequences of spikes spaced either 50 or 49 time steps apart without the help of any short training exemplars [12]. Pascanu, Mikolov and Bengio denotes that LSTMs, to a certain extent, solves the Vanishing Gradient and Exploding Gradient problems faced by other RNNs [13]. Pham, Bluche, Kermorvant, and Louradour claims that the use of dropouts in LSTMs prevent overfitting and improves performance [14]. Refer to Figure 1 for the flow of LSTM in forecasting.

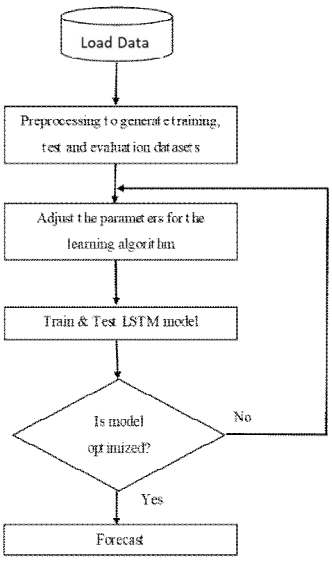

Figure 1 : A General forecasting process using LSTM

Zhou reported that the use of Long Short-Term Memory (LSTM) in stock price forecasting has been done for quite some time and it has achieved some remarkable results [15]. LSTM is programmed to train the computer to best fit the previous data in order to forecast the next value. LSTM learns from the historical data and uses it to train its networks. It tests its predictions against past data to enable it to forecast the future value.

But the problem with classic LSTM is that it does not consider the obscured relationships between the present value, the immediate previous value and the immediate successive value. It does not forecast the next gradient in the time series. LSTM just searches for a pattern that could best fit the whole sequential data without considering the gradients of the immediate past, present and immediate future values. Therefore, a new method that utilizes LTSM 3-point Moving Gradient clusters i.e. the previous, present and future values, is proposed to produce more accurate forecasts. The main goal of this study is to explore the predictability of stock market value direction with LSTM-fed 3-point gradients. The remainder of this paper is organized as follows. In Section 2, we introduce the basic theory of Deep learning. Section 3 gives the experiment scheme. The experiment results are shown in Section 4. Some conclusions are drawn in Section 5.

2. RELATED WORKS

2.1 Deep Learning

The history of Deep Learning began as early as 1943, when Walter Pitts and Warren McCulloch built a computer model based on the human brain's neural networks, mimicking the human thought process via algorithms and Mathematics [16]. Henry J. Kelley then established the fundamentals of a continuous Backward Propagation Model [17] while Stuart Dreyfus established a less complex model based on the chain rule [18]. One of the first few Deep Learning algorithms originates from Alexey Grigoryevich Ivakhnenko , cited in Bansal and Valentin Grigor′evich Lapa in 1965 [19]. Kunihiko Fukushima invented the first "convolutional neural networks" in 1979 [20]. The term Deep Learning was coined by Dechter [21] and introduced by Igor Aizenberg in 2000 [22] . Alexey Ivakhnenko and Lapa invented the concept of multilayer perceptrons in 1965.

Weng recognised that a human brain does not use a massive 3-D object representation and consequently came out with Cresceptron, a self-organizing neural network which expands adaptively [23]. André de Carvalho together with Mike Fairhurst and David Bisset, experimented with a multi-layer Boolean neural network in 1997 and eventually published their findings [24]. Brendan Frey, as cited in Hinton trained a network containing six fully connected layers and several hundred hidden units for 48 hours using the wake-sleep algorithm [25] . A paper by Hochreiter and Schmidhuber on recurrent neural network was published in 1997 [26].

Hinton, Osindero and Teh demonstrated the effective pre-training of a many-layered feedforward neural network at a modest rate of one layer at a time [27]. In 2010, researchers implemented deep learning from TIMIT (a corpus of phonemically and lexically transcribed speech of American English speakers of different sexes and dialects to a speech recognition system armed with a large vocabulary [28].

Giusti, Cireşan, Masci, Gambardella and Schmidhuber demonstrated how max-pooling CNNs on GPU can intensely enhance many vision benchmark records [29]. Giusti, Cireşan, Masci, Gambardella and Schmidhuber 's system also emerged as the winner of the ICPR contest on study of large medical images for detecting cancer [29]. Chong, Han and Park assessed both the advantages and drawbacks of deep learning algorithms for stock market analysis and prediction [30].

2.2. Recurrent Neural Network (RNN)

Donges noted that Recurrent Neural Networks (RNN) are very promising type of neural networks as they are the only that have an internal memory [31],[32]. With the invention of LSTM, which is a newer variant of RNN, and the significant upgrade in computing and memory power these days, RNN has shown its true potential. Because of their internal memory, RNN's can reference its internal memory to have more precise predictions as it considers past values. This reason is why they are tapped for time series forecasting, speech recognition, text translation, financial applications, visual and auditory applications, weather forecasting and others as they can have a better understanding of a sequence. A Recurrent Neural Network (RNN) learns to predict succeeding values based on the present value and the immediate previous value as illustrated in Figure 2.

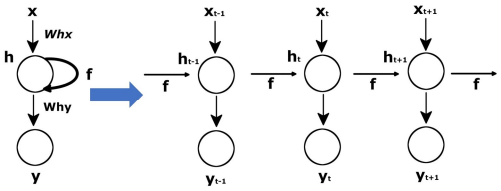

Figure 2 : A Recurrent Neural Network

Past values from an RNN unit is backward propagated to the RNN unit as an additional input along with the regular inputs. FFNNs however do not have feedback from the past and all its data is moving forward.

Thus, an RNN has two sets of inputs – the present and the recent past. RNN applies weights to both the inputs and tweak their weights for both through gradient descent and Backward-Propagation process [31].

Visibly, there is no data being sent back into the FFNN unit as the output is not being considered for tweaking the network. The RNN, as its name suggests, uses its output as feedback to adjust its network for better results.

2.3 Backward-Propagation

According to Donges Backward-Propagation, is the act of sending output data backwards through your neural network to find the partial derivatives of the error and applying appropriate weights to correct it [31]. The fundamentals of nonstop Backward-Propagation were derived in the setting of control theory by Henry J. Kelley [17] and by Arthur E. Bryson who applied the theories of dynamic programming. Stuart Dreyfus constructed a less complex Backward-Propagation derivation solely basing on the chain rule [18]. Bryson and Ho considered Backward-Propagation as a multi-layer dynamic optimization system [33]. Paul Werbos proposed that Backward-Propagation could be used for neural nets in his study [34]. Dreyfus used Backward-Propagation to adjust the parameters of controllers in proportion to error gradients [18]. Werbos suggested the possibility of applying Backward-Propagation to artificial neural networks [34]. Rumelhart, Hinton and Williams demonstrated that Backward-Propagation can produce important internal depictions of incoming data in hidden layers of neural networks [35]. In 1993, Wan E. was the first winner of an international pattern recognition contest using Backward-Propagation [36].

2.4. Gradient Descent

One of the simplest and very well-known methods to find the minimum value of function for unconstrained optimization is gradient method [37]. Gradient descent is an optimization algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient [38]. It does this by finetuning the weights, in an attempt to decreases the error. This is how a Neural Network learns during the training process. Bhattarai explains that Gradient Descent finds the minimum by taking learning steps (rate) proportional to the negative of the gradient of the function at the current point [39]. It edges towards minimum along the steepest gradient. The appropriate size of the learning steps is crucial. If too a large learning step (rate) is taken, we may overshoot and miss the minimum. On the other hand, if a too small learning step is engaged, it may take lots of steps which translates into a long learning time. The lesser the cost function, the more accurate the value prediction is. Ultimately, the process would arrive at the minimum cost function which indicates the most accurate prediction.This way, networks such as LSTM reduces the discrepancy between the predicted value and the actual value to enable it to optimally forecast values accurately.

2.4. Long Short Term Memory (LSTM)

LSTM, according to Olah is a unique type of RNN, able to learn long-term dependencies [40]. Being able to remember information for a long time, LSTMs can avoid long-term dependency problems of other RNNs. LSTM is considered as a type of Recurrent Neural Network (RNN) and an improvement from the classic RNN. The classic RNN faces the problem of vanishing and exploding gradient due to its unrestricted and indiscriminate backward propagation.

2.5. The Vanishing and Exploding Gradient

Alese explains that the gradients produced from the deeper layers proceed through continuous matrix multiplications due of the chain rule, and as they approach the past layers, the small values will shrink exponentially until they vanish [41]. Thus, the model cannot learn, and this is referred as the vanishing gradient problem. Conversely, if they have large values which is more than 1, they get larger and explode crashing the model. This is referred as the exploding gradient problem. Brownlee states that error gradients in RNN, can amass during the Backward-Propagation process resulting in very large gradients [42]. These would bring very large values to the network, resulting in an unstable network. In extreme cases, the values of weights can accumulate and become so enormously that they would overflow, producing NaN values. Soni explains that Hyperbolic tangent(tanh) is mostly is the activation function in RNNs which transform values into the range of 0 and 1 [43]. Backward propagation will cause the gradient to be calculated by chain rule. it can multiply small numbers many times to squeeze the final gradient to almost nil. This would remove the gradient from the weights and stops model training. He further clarifies that conversely to the vanishing gradient problem, following chain rule and multiplying a large value to itself multiple times would lead to an astronomical number which leads to an explosion of gradient. The use of tanh in RNN is unavoidable. It helps to normalize wildly varied values. According to Phi the tanh activation is used to help regulate the values flowing through the network. The tanh function squishes values to always be between -1 and 1 [44]. This RNN problem is solved in LSTM. As Arbel aptly puts it - RNNs suffer from vanishing gradients caused by long series of multiplications of small values, diminishing the gradients and causing the learning process to become degenerate [45]. LSTMs solve the problem by creating a connection between the forget gate activations and the gradients computation, this connection creates a path for information flow through the forget gate for information the LSTM should not forget and discard the information it should forget.

2.5. Classic LSTM

To solve the problem of the Vanishing and Exploding Gradient, Hochreiter and Juergen Schmidhuber, as cited in Singha proposed the LSTM network [46]. This is illustrated in Figure 3.

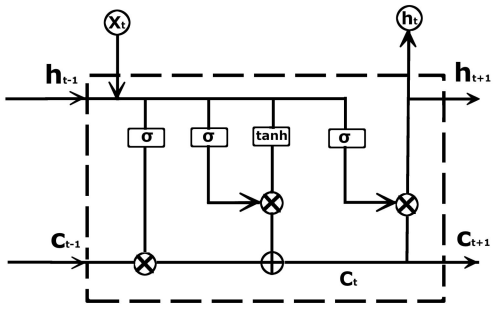

Figure 3: A LSTM Unit

The memory from previous LSTM unit is c_{t-1} while h_{t-1} is the data output from the previous LSTM unit. x_t is the current data input while c_{t+1} is the new updated memory. σ and tanh are the Sigmoid and tanh layers respectively. h_t and c_t are the current data output and memory respectively and they will be forwarded to the next LSTM unit. h_{t+1} is the data output from the current LSTM unit. Sinha states that LSTM can solve the problems of exploding and diminishing gradients, because it uses gates to control the memorizing process [46]. To overcome the vanishing gradient problem, tanh gate is utilised as a suitable function whose second derivative can sustain for a long range before going to zero. The Sigmoid σ gate can output 0 or 1 as it can be used to forget or remember the information. To forget, the Sigmoid σ gate outputs 0 and to remember 1. A LSTM serial network is illustrated in Figure 4.

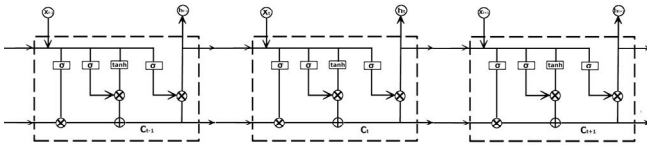

Figure 4: A LSTM Serial Network

LSTMs are connected serially. The data and memory from the previous LSTM are forwarded to next, thus forming a serial chain of LSTM layers. Sinha concluded that "LSTM outperforms the other models when we want our model to learn from long term dependencies. LSTM's ability to forget, remember and update the information pushes it one step ahead of RNNs. Singh noted that LSTM has proven to be extremely effective and has outshone other conventional algorithms in sequence prediction problems application [47]. Goyal, Krishnamurthy, Kulkarni, Kumar, Vartak & Lanham opined that "LSTM Neural Network is one such model that performs better (lower MAPE) than other baseline models and is much simpler to implement as it requires minimal feature engineering [48]. Faisal and Tuga developed the Long Short Term Memory (LSTM) network in studying the problem of time series prediction on the use of drugs in prescribing patients in hospitals [49]. However, with more

data points in each time series the LSTM will train better, and the accuracy of forecasts will be enhanced. The researchers above deemed that LSTM to be the most efficient network for forecasting. However, there are a few drawbacks in the classic LSTMs. Firstly, they are not hardware friendly. It takes lots of resources to train LSTMs fast. RNN and LSTM are difficult to train because they require memory-bandwidth-bound computation, which is the worst nightmare for hardware designer and ultimately limits the applicability of neural networks solutions [50]. Imran noted that LSTMs take longer and require more memory to train. LSTMs are easy to overfit and Dropout is much harder to implement in LSTMs [51]. Lastly, LSTMs are sensitive to different random weight initializations**.**

3. MATERIALS AND METHODS

This study uses Deep Learning to forecast the daily end-of-day stock prices of KLSE (Kuala Lumpur Stock Exchange) Main Market. It will make use of the final selling and buying prices programmatically downloaded from a stock market website. It will use deep learning on a new variant of LSTM recurrent network to forecast prices. The LSTM will forecast using Linear Regression of moving 3-point prices instead of the actual prices themselves. By forecasting the gradient, the next value can be easily calculated.

The purpose of this study is to find out if a stock price forecast be done using LSTM of moving 3-point Linear Regression gradients It is also to find out the accuracy of this LSTM method forecast compared to a normal LSTM forecast. This study hopes to develop a fairly reliable software system be developed using this method.

This study is based on the use of Three Point Moving Gradients. The rationale for the use of 3-point Moving Gradient is the price of a current stock counter price does not differ much from its immediate previous values under normal circumstances. Likewise, its price of the next day cannot differ much from the present-day price. Except in very rare anomalies, changes in stock prices are gradual and in continuum. Therefore, assuming the 3 price points (yesterday, today and tomorrow) are linear, reversed Linear Regression is used to calculate the next price point. Thus, the 3-day prices are used in Linear Regression model.

Linear Regression is a statistical method that explains the relationship between two continuous variables in a simple linear equation as shown in Equation 1.

$$y = bx + c \qquad (1)$$

where ;

x = independent variable

y = dependent variable.

b = gradient

c = initial value of y when x = 0

In stock forecasting, y is the stock price while x refers to the time (day). Given in equation 2,

$$S_{xy} = n\Sigma xy - \Sigma x\Sigma y \qquad (2)$$

$$S_{xx} = n\Sigma x^2 - (\Sigma x)^2$$

where n refers to the number of values and

$$b = \frac{S_{xy}}{S_{xx}} \qquad (3)$$

Regression Line Equation is:

$$y = \bar{y} - b\bar{x}$$

where $\bar{y}$ *and* $\bar{x}$ are mean y and mean x respectively.

The regression line is plotted and extended for the next x and the next y value can be predicted. In this study, LSTM is used to predict the 3-point gradient of the last prices. From this gradient, reversed Linear Regression is used to calculate the next last price. Using the last two points and forecasted gradient, reversed LR method is used to calculate the third point – that is the calculated price. Reverse Linear Regression analysis will determine where the price is heading – upward, downward or remain unchanged. By inputting moving 3-point Linear Regression gradients into LSTM to be trained and tested against historical data, these networks can forecast the next gradient. Consequently, from this gradient and its mean, the next price can be calculated through reversed Linear Regression method.

The gradient forecasted can be either positive or negative. This is different from only positive values in the LSTM forecasting of the next price because prices can never be negative. The next price can be calculated from either positive or negative forecasted gradient. Hence, the conceptual model of the Enhance LSTM with a 3-point Gradient method in stock price forecasting is illustrated as in Figure 5. The last price will be fed into the LSTM network and into past gradients calculation. The past gradients will then be fed into the LSTM Gradient network. From the forecasted next gradient, the next price is calculated. The prices from both purely LSTM forecasts and the calculated next price is compared via RMSE. Both prices are then compared using RMSE to see which is closer to the observed price. Figure 5 shows the conceptual model of enhanced LSTM with 3-point gradient method.

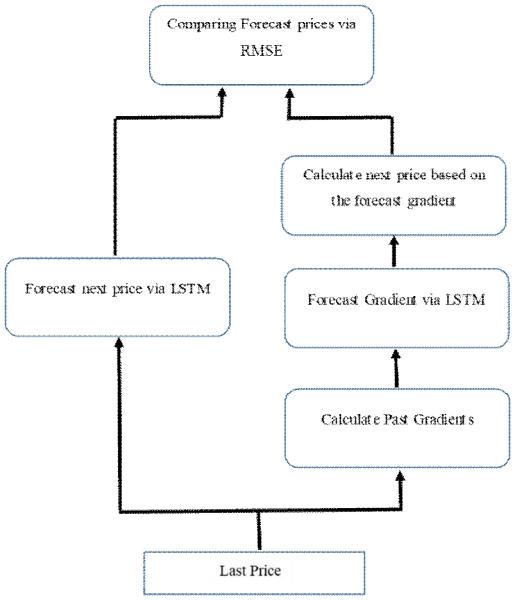

Figure 5: The Conceptual Model of Enhanced LSTM with 3-point Gradient method.

3. RESULTS AND DISCUSSION

The analysis was conducted on the data of three KLSE stock counters dating from 13th May 2019 till 14th October 2019. Three graph lines were plotted are the actual last price, LSTM and the TPMG.

Both the last price and gradient data go through the same LSTM network settings except LSTM for the last price is calculated with sigmoid activation while the LSTM for gradient (TPMG) is calculated with tanh activation. LSTM for the last price is forecasted with sigmoid activation because its forecasted data is only positive values, being forecasted prices. LSTM for gradient (TPMG) is forecasted with tanh activation because its forecasted data has both positive and negative values, being forecasted gradients. The other same settings are optimizer = adam, loss = mean square error and three successive dropouts of 0.2. RMSE (Root Mean Square Error) are then calculated for both LSTM and TPMG to measure how spread out each against the actual last price.

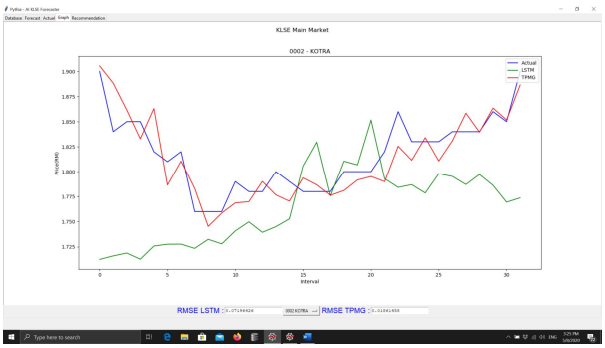

Figure 6: Actual Price-LSTM-TPMG Graph for counter 0002

The result for the stock 0002 is as Figure 6. RMSE LSTM is 0.0796426 while RMSE TPMG is 0.01861458. This shows that TPMG data spread is closer to the actual price than LSTM.

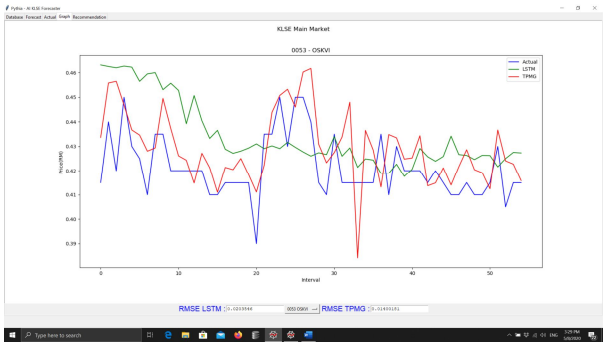

Figure 7: Actual Price-LSTM-TPMG Graph for counter 0053

Similarly, RMSE LSTM for the stock 0053 is 0.0203546 while RMSE TPMG is 0.01400181. It also shows that TMPG data spread is closer resembling the actual data.

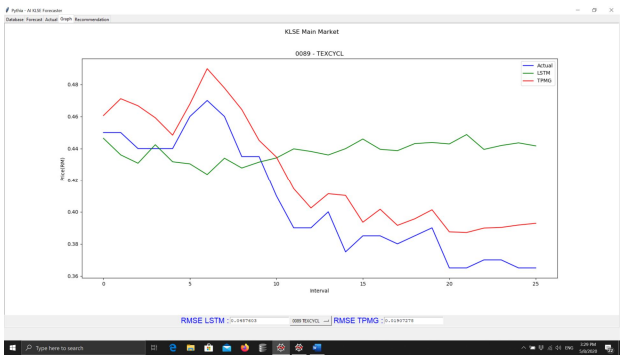

Figure 8: Actual Price-LSTM-TPMG Graph for counter 0089

The RMSE LSTM for the stock 0089 is 0.0487603 while the RMSE TPMG is 0.01907278. This also shows that the data spread for TPMG is closer to the actual data compared to LSTM data

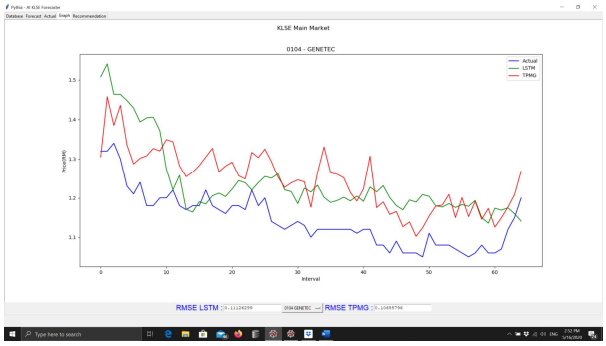

Figure 9: Actual Price-LSTM-TPMG Graph for counter 0104

The RMSE for LSTM for the stock 0104 is 0.11126299 while the RMSE for TPMG is 0.10685796. This shows that TPMG data spread is closer to the actual data than the LSTM.

Table 1: Summary of The Preliminary Results

Dataset Counter #	RMSE Test Set	
	LSTM	TPMG
0002 (From 17.07.2019 To 14.10.2019)	0.0796426	0.01861458
0053 (From 27.06.2019 To 14.10.2019)	0.0203546	0.01400181
0089 (From 17.07.2019 To 10.10.2019)	0.0487603	0.01907278
0104 (From 19.06.2019 To 11.10.2019)	0.11126299	0.10685796

In summary, in all tested cases, TPMG data spread is closer to the actual observed data than LSTM data spread as shown in Table 1. Thus, TPMG predicts data more accurate than LSTM.

4. CONCLUSION

This new method of using moving 3- Point Moving Gradients in LSTM is expected to produce a comparable forecast compared to the classic LSTM if not better as the immediate past and present values are considered to produce a future forecast value. The enhancement in the 3-PMG may be necessary to produce better results.

ACKNOWLEDGEMENTS

The authors would like to thank to Universiti Teknikal Malaysia Melaka (UTeM) and Center for Advanced Computing Technology (C-ACT, Fakulti Teknologi Maklumat dan Komunikasi, UTeM, who provided support and resources for this study.

REFERENCES

1. S. Lahmiri. **Intraday stock price forecasting based on variational mode decomposition.** *Journal of Computational Science,* pp. 23-27, Jan. 2016.
2. L. Kramer. ***How is a company's stock price and market cap determined?****, Investopedia,* 2019. Retrieved June 12, 2019.
3. P. Pandey**.** ***Predicting the 'Future' with Facebook's Prophet.*** Towards Data Science, March 2019. Retrieved June 12, 2019, from https://towardsdatascience.com/predicting-the-future-with-facebook-s-prophet-bdfe11af10ff.
4. P. Brockwell. ***Introduction to Time Series and Forecasting. Second, Foundations.*** *Available : http://www.ccs.fau.edu/~bressler/EDU/EPhys/References/B_D_In.pdf*, 2002. Retrieved June 14, 2019, from http://www.ccs.fau.edu/~bressler/EDU/EPhys/References/B_D_In.pdf
5. R. Torres. ***7 Ways Time Series Forecasting Differs from Machine Learning***, May 2018. Retrieved June 13, 2019 from https://www.datascience.com/blog/time-series-forecasting-machine-learning-differences
6. D. Shah, H. Isah, F. Zulkernine ***Stock Market Analysis: A Review and Taxonomy of Prediction Techniques***. Int. J. of Financial Studies, 7(2), May 2019.
7. D. Brock.. ***The Problem With Forecasting***, January 2013 Retrieved June 13, 2019, from https://customerthink.com/the_problem_with_forecasting
8. C. Rackauckas. ***What is the advantage of using RNN and LSTM over traditional methods for time series of streaming data?****, ResearchGate*, April 2018. Retrieved June 13, 2019, from https://www.researchgate.net/post/What_is_the_advantage_of_using_RNN_and_LSTM_over_traditional_methods_for_time_series_of_streaming_data
9. M. Muller. ***Neural network - Why are RNN/LSTM preferred in time series analysis and not other NN? .*** Data Science Stock Exchange, September 2017. Retrieved June 13, 2019, from https://datascience.stackexchange.com/questions/23029/why-are-rnn-lstm-preferred-in-time-sreries-analysis-and-not-other-nn/
10. V. Calomme. ***Why are RNN/LSTM preferred in time series analysis and not other NN?*** September 2017. Retrieved from Data Science: https://datascience.stackexchange.com/questions/23029/why-are-rnn-lstm-preferred-in-time-series-analysis-and-not-other-nn/23076
11. D. Cabrera, A. Guaman, S. Zhang, M. Cerrada, R.V. Sanchez, J. Cevallos, J.Y. Long & C**.** Li. **Bayesian approach and time series dimensionality reduction to LSTM-based model-building for fault diagnosis of a reciprocating compressor**. Neurocomputing, pp. 51-66, September 2019. Retrieved August 11, 2020
12. F.A. Gers, J. Schmidhuber and F**.** Cummins. **Learning to Forget: Continual Prediction with LSTM.** ***Neural Computation, 12*(10),** *MIT Press Journal***,** pp. 2451-2471, 2000.
13. R. Pascanu, T. Mikolov and Y. Bengio. ***On the difficulty of training recurrent neural networks***, 2013. Retrieved JUNE 15, 2019, from http://proceedings.mlr.press/v28/pascanu13.pdf
14. V. Pham, T. Bluche, C. Kermorvant, and J. Louradour. ***Dropout improves Recurrent Neural Networks for Handwriting Recognition***, 2013. Retrieved June 19, 2019, from https://arxiv.org/pdf/1312.4569.pdf
15. C. Zhou, C. Sun, Z. Liu and F. Lau**. A C-LSTM Neural Network for Text Classification,** November 2015. Retrieved June 19, 2019, from https://www.researchgate.net/publication/285458953_A_C-LSTM_Neural_Network_for_Text_Classification

16. W.S. McCulloch and W. Pitts. **A logical calculus of the ideas immanent in nervous activity.** Bulletin of Mathematical Biophysics 5, pp.115–133, Dec. 1943.
17. H. J. Kelley. **Gradient Theory of Optimal Flight Paths**. *ARS Journal, 30*(10), pp. 947-954, Oct. 1960.
18. S. Dreyfus. **The numerical solution of variational problems**. *Journal of Mathematical Analysis and Applications, 5*(1), pp. 30–45, Aug. 1962.
19. J.C. Bansal, K.N. Das, A. Nagar, , K. Deep and A.K. Ojha. ***'Soft Computing for Problem Solving: SocProS 2017'***. New York: Springer, 2017.
20. K. Fukushima. ***Artificial Vision by Deep CNN Neocognitron,*** 1979. Retrieved from http://ias.ust.hk/events/201612nbni/doc/Abstract_Fukushima_Kunihiko.pdf
21. R. Dechter. ***Learning While Searching in Constraint-Satisfaction-Problems***. 1986. Retrieved June 12, 2019, from https://www.aaai.org/Papers/AAAI/1986/AAAI86-029.pdf
22. I. Aizenberg. ***Multi-Valued and Universal Binary Neurons: New Solutions in Neural Networks***, 2001. Retrieved June 20, 2019, from https://www.academia.edu/2656424/Multi-Valued_and_Universal_Binary_Neurons_New_Solutions_in_Neural_Networks
23. J. Weng, N. Ahuja and T.S**.** Huang **Learning Recognition and Segmentation Using the Cresceptron**. *International Journal of Computer Vision 25,* pp. 109–143, Nov. 1997.
24. A. Calvalho, P. D. L. Ferriera, M.C. Fairhurst & D.L. Bisset. **Implementation of boolean neural networks on parallel computers,** *II Congreso Argentino de Ciencias de la Computación*, pp. 133-145. 1996.
25. G. E. Hinton. ***The wake-sleep algorithm for unsupervised neural networks,*** April 1995. Retrieved June 29, 2019, from https://www.cs.toronto.edu/~hinton/csc2535/readings/ws.pdf
26. S. Hochreiter. **The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions**. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 6*(2), pp. 107-116, April 2016.
27. G. E. Hinton, S. Osindero and Y. Teh ***A Fast Learning Algorithm for Deep Belief Nets***. Neural Computation, July 2006.
28. J.S. Garofolo, L.F. Lamel, W.M. Fishe, J.G. Fiscus, D.S. Pallett and N.L. Dahlgren. ***TIMIT Acoustic-Phonet,ic Continuous Speech Corpus.*** February 1993. Retrieved June 19, 2019 from https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nistir4930.pdf
29. A. Giusti, D. C. Cireşan, J. Masci, L. M. Gambardella and J. Schmidhuber, **Fast image scanning with deep max-pooling convolutional neural networks**, *IEEE International Conference on Image Processing*, Melbourne, Feb. 2013.
30. E. Chong, C. Han & F.C. Park. ***Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies***. Expert Systems with Applications, October 2017. Retrieved June 20, 2019, from https://doi.org/10.1016/j.eswa.2017.04.030
31. N. Donges. ***A Guide to RNN: Understanding Recurrent Neural Networks and LSTM.*** Builtin, June 2019. Retrieved June 11, 2019, from https://builtin.com/data-science/recurrent-neural-networks-and-lstm
32. S. A. Asmai, B. Hussin, M. M. Yusof & A. S. Shibghatullah. **Time Series Prediction Techniques for Estimating Remaining Useful Lifetime of Cutting Tool Failure**. *International Review on Computers and Software*, vol 9. No.6. 2014.
33. A.E. Bryson, Y. C. Ho. ***Applied Optimal Control : Arthur E. Bryson****, Jr.,* Blaisdell, 1975.
34. P. J. Werbos. ***Backwards Differentiation in AD and Neural Nets: Past Links and New Opportunities.*** National Science Foundation, 2006.
35. D. Rumelhart, G. Hinton and R. Williams. **Learning representations by back-propagating errors.** Nature Research Journal, pp. 323, 533–536. Oct. 1986.
36. E. A. Wan. **Time series prediction by using a connectionist network with internal delay lines.** In SANTA FE INSTITUTE STUDIES IN THE SCIENCES OF COMPLEXITY-PROCEEDINGS VOLUME- Vol. 15, pp. 195-195. Addison-Wesley publishing co., 1994.
37. F.H. Siti, M. Mustafa, M.H.I. Asrul,M. Rivaie and M.A. Mohamad. **Modification of Search Direction in Steepest Descent Method and Its Application in Regression Analysis**, *International Journal of Advanced Trends in Computer Science and Engineering,Volume 9, No.1.1*, 2020 https://doi.org/10.30534/ijatcse/2020/1791.12020
38. GitHub. **Gradient Descent**. 2017. Retrieved from GitHub: https://ml-cheatsheet.readthedocs.io/en/latest/gradient_descent.html
39. S. Bhattarai. ***WHAT IS GRADIENT DESCENT IN MACHINE LEARNING?*** A Tech Blog, June 2018. Retrieved June 19, 2019, from https://saugatbhattarai.com.np/what-is-gradient-descent-in-machine-learning/
40. C. Olah. ***Understanding LSTM Networks***. Colah's blog August 2015. Retrieved March 21, 2019, from https://colah.github.io/posts/2015-08-Understanding-LSTMs/
41. E. Alese. ***The curious case of the vanishing & exploding gradient.*** Learn.Love.AI, June 2018. Retrieved June 14, 2019, from https://medium.com/learn-love-ai/the-curious-case-of-the-vanishing-exploding-gradient-bf58ec6822eb
42. J. Brownlee. ***A Gentle Introduction to Exploding Gradients in Neural Networks***,. Machine learning Mastery, Dec. 2017.

43. M. Soni. ***Understanding architecture of LSTM cell from scratch with code***. June 2018. Retrieved June 14, 2019, from https://hackernoon.com/understanding-architecture-of-lstm-cell-from-scratch-with-code-8da40f0b71f4
44. M. Phi. **Illustrated Guide to LSTM's and GRU's: A step by step explanation**. Retrieved June 14, 2019 from https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21
45. N. Arbel. ***How LSTM networks solve the problem of vanishing gradients,***. Data Driven Investor, December 2018. Retrieved June 14, 2019, from https://medium.com/datadriveninvestor/how-do-lstm-networks-solve-the-problem-of-vanishing-gradients-a6784971a577
46. N. Sinha. ***Understanding LSTM and its quick implementation in keras for sentiment analysis,*** February 2018. Retrieved March 29, 2019, from https://towardsdatascience.com/understanding-lstm-and-its-quick-implementation-in-keras-for-sentiment-analysis-af410fd85b47
47. D. Singh **Artificial Neural Network In Python Using Keras For Predicting Stock Price Movement**, 2018. Retrieved April 5, 2019, from https://www.datacamp.com/community/news/artificial-neural-network-in-python-using-keras-for-predicting-stock-p-091213ndcfcg
48. A. Goyal, R. Krishnamurthy, S. Kulkarni , R. Kumar , M. Vartak and M.A. Lanham. ***A Soluion to Forecast Demand using Long Short Term Memory Recurrent Networks For Time Series Forecasting***. Purdue University, 2018. Retrieved June 19, 2019,
49. B. Faisal and M. Tuga. **Drug Stock Optimization Based on Consumption Patterns for Hospital Formulary Using Deep Learning Approach**, *International Journal of Advanced Trends in Computer Science and Engineering vol 9. No. 3*, June 2020. https://doi.org/10.30534/ijatcse/2020/31932020
50. E. Culurciello. ***The fall of RNN / LSTM***. Towards Data Science, April 2018. Retrieved June 19, 2019, from https://towardsdatascience.com/the-fall-of-rnn-lstm-2d1594c74ce0
51. Imran**. *neural network - So what's the catch with LSTM?*** Data Science Stack Exchange, February 2018. Retrieved June 22, 2019, from https://datascience.stackexchange.com/questions/27392/so-whats-the-catch-with-lstm/27401