**Bulletin of Electrical Engineering and Informatics**
Vol. 12, No. 6, December 2023, pp. 3780~3789
ISSN: 2302-9285, DOI: 10.11591/eei.v12i6.4987

# Adaptation of stochasticity into activation function of deep learning for stock price forecasting

**Assunta Malar Patrick Vincent, Hassilah Salleh**
Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu (UMT), Terengganu, Malaysia

**Article Info**

***Article history:***

Received Oct 11, 2022
Revised Jan 10, 2023
Accepted Feb 23, 2023

***Keywords:***

Forecasting stock price
Gaussian process
Multilayer perceptron
Stochastic neural network

**ABSTRACT**

Stock market is an example of a stochastic environment in the real world. multilayer perceptron (MLP) is often applied to forecast stock price. However, it is widely used to approximate the input-output mapping deterministically. Hence, this study aims to adapt stochasticity into MLP by introducing the Gaussian process into the sigmoid activation function. In addition, the adapted activation function incorporates Roger-Satchell and Yang-Zhang volatility estimators. Besides, the stochastic activation function was considered as a hyperparameter by applying it either only in training time or in both testing and training time. The stochastic multilayer perceptron (S-MLP) is then applied to forecast one day's highest stock price of eight constituents in FTSE Bursa Malaysia KLCI (FBMKLCI). The result shows that the proposed network is inferior in comparison to MLP except for several constituents. In addition, S-MLP with stochastic activation function during both the training and testing time performs better compared to the presence of stochastic activation function in S-MLP during training time only.

***Corresponding Author:***

Hassilah Salleh
Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu
Kuala Nerus, Terengganu 21030, Malaysia
Email: hassilah@umt.edu.my

## 1. INTRODUCTION

Stock markets are associated with financial markets which is a complicated dynamic, non-linear system that generates a lot of noise. Stock markets are influenced by various random events which can be classified into two main categories: political and economic [1]. These influential factors create random events which result in a noisy environment and uncertainty in the stock market. According to Mostafa and Atiya [2] stock prices are classified as one of the "noisiest" time series, which means the stock price fluctuation follows a stochastic process–a noisy, dynamic, non-linear, non-parametric, non-stationary, and chaotic process. Hence, it is challenging to forecast the future value [2]–[4], yet it is possible.

Thus, various mathematical models have been developed for the application of forecasting stock prices. It can be classified into statistical models and machine learning models [5]–[7]. Deep learning techniques, a subset of machine learning [8] are now more popular than ever due to the recent improvement in the field of stock market forecasting [9]. It automatically extracts the relevant features from dense and noisy data, detects the hidden non-linear relationship [10], and approximates all the internal parameters through incremental learning [11]. These properties of the neural network make it appropriate to address some of the challenges associated with the financial market and it creates room to develop a neural network that represents the random process in the stock prices. Multilayer perceptron (MLP) is a class of deep neural networks (DNN), it is widely applied to forecast stock prices. It only approximates the deterministic input and output mapping. Hence, it does not represent the aforementioned characteristic of stock price [11]–[13]. Therefore, an appropriate stochastic process should be incorporated into MLP to represent the stochastic environment of the financial market when it is applied to be forecasted.

- Related research

Among the MLP models that have hybridised stochasticity in the application of forecasting financial markets are stochastic time effective neural network (STNN) [14]; extension of STNN [13], [15]–[18]; and stochastic neural network [12]. Liao and Wang [14] incorporated brownian motion (BM) into the loss function which is used to train the neural network and then applied to forecast the closing price of the next trading day. Wang *et al.* [15] developed jump stochastic time effective neural network (JSTNN), an extension of STNN, to forecast the crude oil price and stock price. The author introduced BM and Poisson jump into the loss function of artificial neural network (ANN). ANN is a predecessor of MLP; however, it is not DNN. Both studies evaluated the neural network by modelling the relationship between the actual data and the forecasted data through a linear regression analysis. From the evaluation, it is found that the relationship of both the data indicated a strong positive relationship. Neither of these studies performed a comparative analysis between the deterministic counterpart, ANN. Wang and Jun [17] developed an extension of STNN by incorporating principle component analysis (PCA), STNN-PCA to forecast the stock price of Shanghai stock exchange, HS3000, SP500, and Dow Jones Industrial Average. The author conducted a comparative analysis between ANN and STNN, it was concluded that STNN outperformed ANN. Lastly, Jay *et al.* [12] proposed SNN to predict the price of cryptocurrency by adapting random walk theory into the activation function of the neural network. The proposed activation function was applied on MLP and long short-term memory (LSTM). It was found that the proposed model performs better than the deterministic MLP and LSTM when the perturbation factor in the SNN is trained via gradient decent backpropagation algorithm.

All the studies state that the stochasticity was introduced into the deterministic neural network. So, it can mimic and adapt to the trend of the financial market without changing the original trend. Apart from that, it can be deduced that the adaptation of stochasticity into MLP improves the accuracy of forecasting. According to Jay *et al.* [12], there are two ways to incorporate stochasticity into a neural network. Firstly, by randomly changing the weights by a small degree and the author emphasised that it is not ideal as it would mean that the feature detection will eventually get noisy and the network forgets the dependencies. The second method would be by adding randomness into the activation function, it can be viewed as mimicking the noisy characteristics of the financial market [12]. Lastly, the presence of stochasticity allows the loss function to escape the saddle points during the optimization of a neural network when it is trained and it can reach the global minimum [19].

- Research gap and motivation

Gulcehre *et al.* [19] proposed to train the neural network with a strongly saturated activation function by incorporating noise to model a stochastic activation function. The author emphasized that the introduction of appropriate noise will allow the gradient to flow steadily. The author considered the Gaussian process to be incorporated into the saturated activation function and applied it to various neural networks for a different type of dataset. However, the authors did not apply it on forecasting stock prices. Besides, the author incorporated noise into the hard-sigmoid activation function, not into the sigmoid activation function. Furthermore, considered stochastic activation function only during training time, therefore, this study will also experiment on the presence of stochastic activation function during testing time as well [19].

Motivated by the aforementioned discussion, in this paper, we provide a prediction model, which can be used to forecast the next day's highest price of eight constituents listed in FTSE Bursa Malaysia KLCI (FBMKLCI). The Gaussian process with the perturbation factor as the volatility estimators proposed by [20], [21] was incorporated into the activation function, to address the stochastic environment of the stock price. Hence, this research contributes to the mathematical formulation of the stochastic activation function into MLP by representing the characteristic of respective stock prices. Followed by the proposal of a stochastic multilayer perceptron (S-MLP) algorithm with a different number of epochs for stock price forecasting.

## 2. METHOD

This section discusses the architecture of MLP applied in this research. Then, the incorporation of stochasticity into sigmoid activation function to develop S-MLP. Finally, this section ends with the forecasting algorithm of proposed S-MLP and the experimental setup.

### 2.1. Multilayer perceptron

MLP has the ability to efficiently learn and represent the linear and non-linear dependencies between the input and output variables [12]. A standard MLP is made up of three main layers: input layer, output layer, and arbitrary number of hidden layers. The input layers are fully connected to a system of hidden layers by intermediary layers, which then transmit the trained data to the output layers. Each layer is fully connected with weight and biases, then the activation function is applied after the linear matrix computation. These functions squash the output of each hidden layer into a range based on the selected activation function. Backpropagation is a learning algorithm widely used to train the neural network. During the training process, randomly initialized value of weights and biases are iteratively updated by minimizing the mean square error (MSE) and propagating error signals as gradients of the weight and biases [12], [22]. In this study, the neural networks are trained via a backpropagation algorithm and stochastic gradient descent (SGD) and adaptive moment estimation (ADAM) optimizers are used. The geometric pyramid rule proposed by [23] was used to build the architecture of MLP as illustrated in Figure 1 and (1) to (3):

$$h^{(1)} = \Phi^{(1)}\left(W^{(1)}x + b^{(1)}\right) \tag{1}$$

$$h^{(2)} = \Phi^{(2)}\left(W^{(2)}h^{(1)} + b^{(2)}\right) \tag{2}$$

$$\hat{y} = W^{(3)}h^{(2)} + b^{(3)} \tag{3}$$

Where $x \in \mathbb{R}^5$ is input vector; $\hat{y} \in \mathbb{R}^1$ is output vector; $h^{(1)} \in \mathbb{R}^4$ and $h^{(2)} \in \mathbb{R}^2$ are the outputs of the first and second hidden layers respectively; $\mathrm{W}^{(1)} \in \mathbb{R}^{4\times5}$, $\mathrm{W}^{(2)} \in \mathbb{R}^{2\times4}$, and $\mathrm{W}^{(3)} \in \mathbb{R}^{1\times2}$ are weight matrices associated to first, second hidden layer, and output layer; $b^{(1)} \in \mathbb{R}^4$, $b^{(2)} \in \mathbb{R}^2$, and $b^{(3)} \in \mathbb{R}$ are bias vector associated to first, second hidden layer, and output layer; and $\Phi^{(1)} \in \mathbb{R}^4$ and $\Phi^{(2)} \in \mathbb{R}^2$ are activation function in first and second hidden layer.

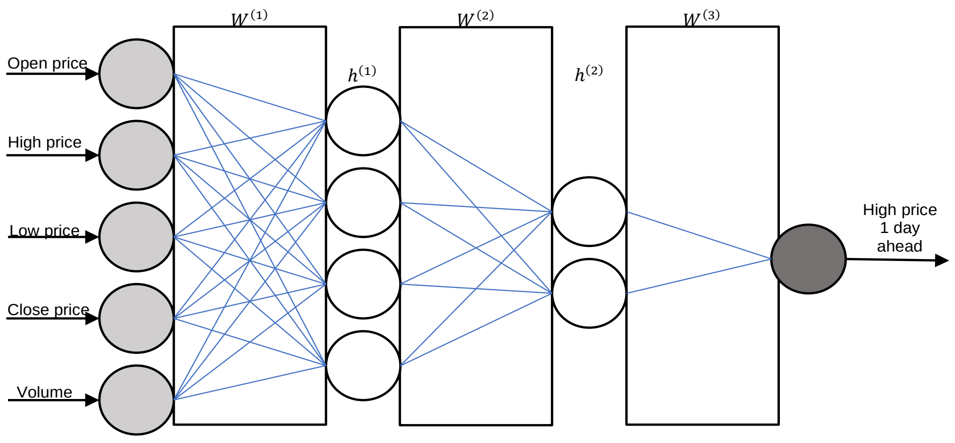

Figure 1. Neural network architecture used to forecast stock price

## 2.2. Activation function

In this study Gaussian process is introduced into the sigmoid activation function. Sigmoid activation function is shown in (4) and it maps the whole real range of $z$ into [0,1] in the $\Phi(z)$ as shown in Figure 2 [24].

$$\Phi(z) = \frac{1}{1+e^{-z}} \tag{4}$$

$\Phi(z)$ refers to sigmoid activation function, then the stochastic sigmoid activation function which follows a Gaussian process can be written as (5):

$$\overline{\Phi}(z,\xi) = \Phi(z) + s,\ s = \mu + \hat{\sigma}\xi \tag{5}$$

Where $\overline{\Phi}(.)$ is sigmoid stochastic activation function; $\xi$ is an 'noise' variable that produces a vector of independent and identically distributed random variable with the same dimension as the activation function and it follows Gaussian process; $s$ is Gaussian process which is normally distributed with mean zero and variance one, $\mu$ is the mean of the stochastic process, $s$; and $\hat{\sigma}$ is the volatility of the stochastic process, $s$, and it is also referred to as the perturbation factor that controls the amount of stochasticity.

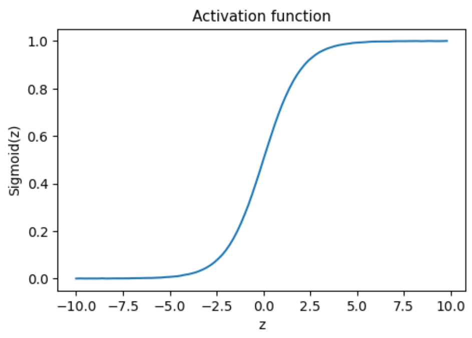

Figure 2. Sigmoid function *Φ*(z) [25]

It is stated that the desirable property of a stochastic activation function is that the activation function should be equal to the respective deterministic activation function. Therefore, the expectation should be approximate to the deterministic as shown in (6) [19]. Since the $\xi$ is a Gaussian noise which is normally distributed with mean zero and volatility one. To achieve the desirable property of the stochastic activation function $\mu$ is assumed to be zero.

$$\mathrm{E}_{\xi \sim N(0,1)}[\overline{\Phi}(z,\xi)] \approx \Phi(z) \tag{6}$$

The novel idea behind the proposed stochastic activation function is the added noise could represent the stochastic environment of the respective stock price when applied to MLP to forecast the stock price. Thus, this study applies volatility proposed by [20], [21] as shown in (7) and (8), respectively instead of the scaled volatility utilized in [19]. Both these estimators were chosen in this experiment because they consider four out of 5 input features of the neural network which are daily opening, high, low, and closing prices. By considering these volatility estimators as the perturbation factor, the changes or volatility of the stock price of the selected constituents will be represented in the neural network.

$$\hat{\sigma}^2{}_{RS} = \frac{1}{n}\sum_{t=1}^{n} h_t(h_t - c_t) + l_t(l_t - c_t) \tag{7}$$

$$\hat{\sigma}^2{}_{YZ} = \hat{\sigma}^2{}_o + k\hat{\sigma}^2{}_c + (1-k)\hat{\sigma}^2{}_{RS} \tag{8}$$
$$k = \frac{\alpha - 1}{\alpha + \frac{n+1}{n-1}}$$

Yang and Zhang [21] suggested to use $\alpha$=1.34 hence $k = \frac{0.34}{1.34+\frac{n+1}{n-1}}$. $\hat{\sigma}^2{}_o$ and $\hat{\sigma}^2{}_c$ were estimated using the classical variance estimator using opening and closing price respectively, represented in (9) and (10):

$$\hat{\sigma}^2{}_o = \frac{1}{n-1}\sum_{t=1}^{n}(o_t - \bar{o})^2, \ \bar{o} = \frac{1}{n}\sum_{t=1}^{n} o_t \tag{9}$$

$$\hat{\sigma}^2{}_c = \frac{1}{n-1}\sum_{t=1}^{n}(c_t - \bar{c})^2, \ \bar{c} = \frac{1}{n}\sum_{t=1}^{n} c_t \tag{10}$$

Where $O_t$ is opening price on date $t$; $C_t$ is closing price on date $t$; $H_t$ is highest price on date $t$; $L_t$ is lowest price on date $t$; $o_t = \ln O_t - \ln C_{t-1}$ is the normalized opening price; $c_t = \ln C_t - \ln O_t$ is the normalized closing price; $h_t = \ln H_t - \ln O_t$ is the normalized highest price; and $l_t = \ln L_t - \ln O_t$ is the normalized lowest price.

### 2.3. Stochastic multilayer perceptron

The incorporation of stochastic sigmoid activation function into MLP yields S-MLP. The proposed S-MLP was applied to forecast the next day's highest stock price. The model was built using python TensorFlow and Keras library. Algorithm 1 illustrates the algorithm from S-MLP along with the tuning on the number of epochs.

```
Algorithm 1 S-MLP to forecast stock price
Input 1: Stock price with daily open, high low, close price and volume.
Input 2: $\mathrm{X}_t$, Daily stock prices (open, high low, close) and volume $t \in \{1, \dots, n\}$.
Input 3: $\mathrm{A}_{t+1}$, Actual daily highest price which has been scaled one day ahead $t \in \{1, \dots, n\}$ .
Input 4: $\hat{\sigma}_{RS}$ and $\hat{\sigma}_{YZ}$.
Output: $\hat{y}_{t+1}$, Daily 1 day ahead forecasted highest price $t \in \{1, \dots, n\}$.
        $\hat{\sigma} = \hat{\sigma}_{RS}, \hat{\sigma}_{YZ}$
        $\mathrm{X} = \mathrm{X}_t$
        $\mathrm{A} = \mathrm{A}_{t+1}$
        $\overline{\Phi}(z, \xi) = \Phi(z) + \hat{\sigma}\xi$
        Number of epochs, $E$
Forecasting with SNN:
        $while\ E < condition$
                Model: $\mathrm{h}^{(1)} = \overline{\Phi}^{(1)}\left(W^{(1)T}\mathrm{X} + \mathrm{B}^{(1)}\right)$
                        $\mathrm{h}^{(2)} = \overline{\Phi}^{(2)}\left(W^{(2)T}\mathrm{h}^{(1)} + \mathrm{B}^{(2)}\right)$
                        $\hat{y} = W^{(3)T}\mathrm{h}^{(2)} + \mathrm{B}^{(3)}$
                Optimize the model: *model.compile*
                Train the model with $E$: *model.fit*
                Obtain the loss value: *model.evaluate*
                Predict the daily highest price: *model.predict*
                $E$+= desired increment
                        $if\ \ R^2 >$ desired accuracy *then*
                        $end\ if$
        $end\ while$
```

### 2.4. Experimental setup

Daily historical prices: opening, high, low, closing prices, and volume of eight constituents listed in FBMKLCI from 6 March 2017 to 25 March 2022 were collected from the Yahoo finance search engine. Among them are Inari Amertron Bhd (0166.KL); Nestle (Malaysia) Berhad (4707.KL); Genting Malaysia Berhad (4715.KL); Hartalega Holdings Berhad (5168.KL); PBB Group Berhad (4065.KL); Malayan Banking Berhad (1155.KL); Top Glove Corporation Berhad (7113.KL); and Kuala Lumpur Kepong Bhd (2445.KL). Then, the collected data is then cleaned to remove NAN values and then it is normalized using min-max normalization. The normalized data were then split into 90% of training set and 10% of testing set. The trained models are evaluated based on the following different performance metrics. MSE, root mean square error (RMSE), mean absolute error (MAE), and coefficient of determination ($R^2$) as shown in (11) to (14) respectively [26]:

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(A_t - \hat{y}_t)^2 \tag{11}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(A_t - \hat{y}_t)^2} \tag{12}$$

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|A_t - \hat{y}_t| \tag{13}$$

$$R^2 = 1 - \frac{\sum_{t=1}^{n}(\hat{y}_t - A_t)^2}{\sum_{t=1}^{n}(A_t - \bar{A}_t)^2} \tag{14}$$

where $A_t$ is the actual price and $y_t$ is forecasted price.

Among the hyperparameters considered in this experiment are: i) optimizers: ADAM and SGD; ii) number of epochs: 500 to 14,500 with an increment of 500 each forecasting; iii) perturbation factor in $\overline{\Phi}(z, \xi)$: $\hat{\sigma}_{RS}$ and $\hat{\sigma}_{YZ}$; and iv) the presence of $\overline{\Phi}(z, \xi)$ during testing time. In the absence of stochastic sigmoid activation function, $\overline{\Phi}(z, \xi)$, during the testing time, S-MLP uses the sigmoid activation function, $\Phi(z)$, to predict the one-day ahead highest stock price. The experiment was conducted with just a drop and replacement of the activation function in existing experimental setups without any changes in the hyperparameters which were set previously.

## 3. RESULTS AND DISCUSSION

This research provides experimental results of MLP, $\overline{\Phi}(z,\xi)$ with $\hat{\sigma}_{RS}$ and $\overline{\Phi}(z,\xi)$ absent during the testing time [S-MLP 1]; with $\hat{\sigma}_{YZ}$ and $\overline{\Phi}(z,\xi)$ absent during the testing time [S-MLP 2]; with $\hat{\sigma}_{RS}$ and $\overline{\Phi}(z,\xi)$ present during the testing time [S-MLP 3]; and with $\hat{\sigma}_{YZ}$ and $\overline{\Phi}(z,\xi)$ present during the testing time [S-MLP 4] when the neural network is optimized with SGD and ADAM. To obtain a more accurate result, the number of epochs was fine-tuned with an increment of 500. To compare and analyze the accuracy of each model, this study extracts the neural network with the number of epochs which have given the highest accuracy and $R^2$. Thus, the number of epochs for each case differs as summarized in Table 1.

Table 1. Number of epochs for all the experimental set up with the highest accuracy

| Optimizer | Model | Constituents | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0166.KL | 4707.KL | 4715.KL | 5168.KL | 4065.KL | 1155.KL | 7113.KL | 2445.KL |
| SGD | MLP | 14,500 | 12,000 | 13,500 | 3,500 | 3,500 | 14,000 | 14,500 | 12,500 |
| | S-MLP 1 | 5,000 | 2,000 | 10,500 | 2,500 | 12,500 | 10,000 | 9,000 | 11,500 |
| | S-MLP 2 | 6,500 | 5,000 | 9,500 | 9,500 | 500 | 6,000 | 8,500 | 12,000 |
| | S-MLP 3 | 9,500 | 1,000 | 4,500 | 13,000 | 7,500 | 7,000 | 10,000 | 11,000 |
| | S-MLP 4 | 11,500 | 13,000 | 10,500 | 13,500 | 1,000 | 14,000 | 13,500 | 12,000 |
| ADAM | MLP | 14,000 | 2,000 | 14,500 | 4,500 | 3,000 | 10,500 | 14,000 | 12,000 |
| | S-MLP 1 | 13,000 | 14,000 | 2,500 | 7,000 | 5,500 | 4,000 | 10,000 | 13,000 |
| | S-MLP 2 | 9,500 | 2,000 | 1,500 | 13,500 | 4,000 | 8,500 | 8,500 | 10,500 |
| | S-MLP 3 | 14,500 | 7,500 | 12,500 | 4,500 | 5,000 | 12,000 | 6,000 | 11,500 |
| | S-MLP 4 | 12,000 | 5,500 | 14,500 | 10,000 | 1,500 | 6,500 | 9,000 | 13,000 |

$R^2$ indicates how much the target values are being captured by the model. Therefore, the respective value of $R^2$ resulted by each model with the number of epochs represented in Table 1 is summarized in Table 2 for the neural network trained using SGD and ADAM. $R^2$ usually shows the percentage variance that the independent variables accounted for in the dependent variable's variance. Moreover, $R^2$ is bounded to the range of $(-\infty, 1)$. It is stated that when the value of $R^2$ approaches the upper bound it means that the predictive model results in accurate forecasting and vice versa if $R^2$ approaching zero. However, if $R^2$ results in a negative value, then it strongly suggested that the given predictive model has the worse fit [27], [28]. From Table 2 it can be observed that both MLP and variations of S-MLP models trained with SGD for 0116.KL constituents resulted in negative $R^2$ value. Thus, it absolutely indicates that the considered neural network is the worst predictive model to forecast the next day's highest price for the respective constituent. Meanwhile, the predictive models of the remaining constituents resulted in $R^2$ value of more than 0.5. This demonstrates that both MLP and variations of S-MLP are appropriate to forecast one day ahead highest price.

Table 2. $R^2$ of the selected models with the respective number of epochs

| Optimizer | Model | Constituents | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0166.KL | 4707.KL | 4715.KL | 5168.KL | 4065.KL | 1155.KL | 7113.KL | 2445.KL |
| SGD | MLP | -1.219 | 0.551 | 0.884 | 0.926 | 0.855 | 0.981 | 0.927 | 0.948 |
| | S-MLP 1 | -1.175 | 0.756 | 0.874 | 0.918 | 0.844 | 0.981 | 0.922 | 0.946 |
| | S-MLP 2 | -0.189 | 0.562 | 0.852 | 0.912 | 0.851 | 0.980 | 0.919 | 0.949 |
| | S-MLP 3 | -0.341 | 0.531 | 0.877 | 0.925 | 0.847 | 0.980 | 0.923 | 0.951 |
| | S-MLP 4 | -0.519 | 0.644 | 0.875 | 0.925 | 0.855 | 0.981 | 0.920 | 0.956 |
| ADAM | MLP | 0.726 | 0.732 | 0.887 | 0.926 | 0.853 | 0.982 | 0.934 | 0.949 |
| | S-MLP 1 | 0.854 | 0.659 | 0.879 | 0.905 | 0.848 | 0.981 | 0.930 | 0.951 |
| | S-MLP 2 | 0.864 | 0.756 | 0.878 | 0.923 | 0.842 | 0.981 | 0.930 | 0.952 |
| | S-MLP 3 | 0.880 | 0.701 | 0.886 | 0.917 | 0.838 | 0.982 | 0.926 | 0.956 |
| | S-MLP 4 | 0.845 | 0.722 | 0.880 | 0.926 | 0.850 | 0.981 | 0.899 | 0.952 |

Figures 3 and 4 are the pictorial outcome of the actual value and forecasted value with all the variations of the neural network when it is optimized using SGD and ADAM respectively. S-MLP uses the Gaussian process with the [20], [21] volatility estimators as perturbation factors. This randomness is directly applied to the sigmoid activation function. This results in an internal randomness function. Hence, it helps to escape the local minima during the learning process and reach the global minima [19], [26]. Thus, almost all the variations of S-MLP were able to generate the pattern of the forecasted price almost similar to the actual price.

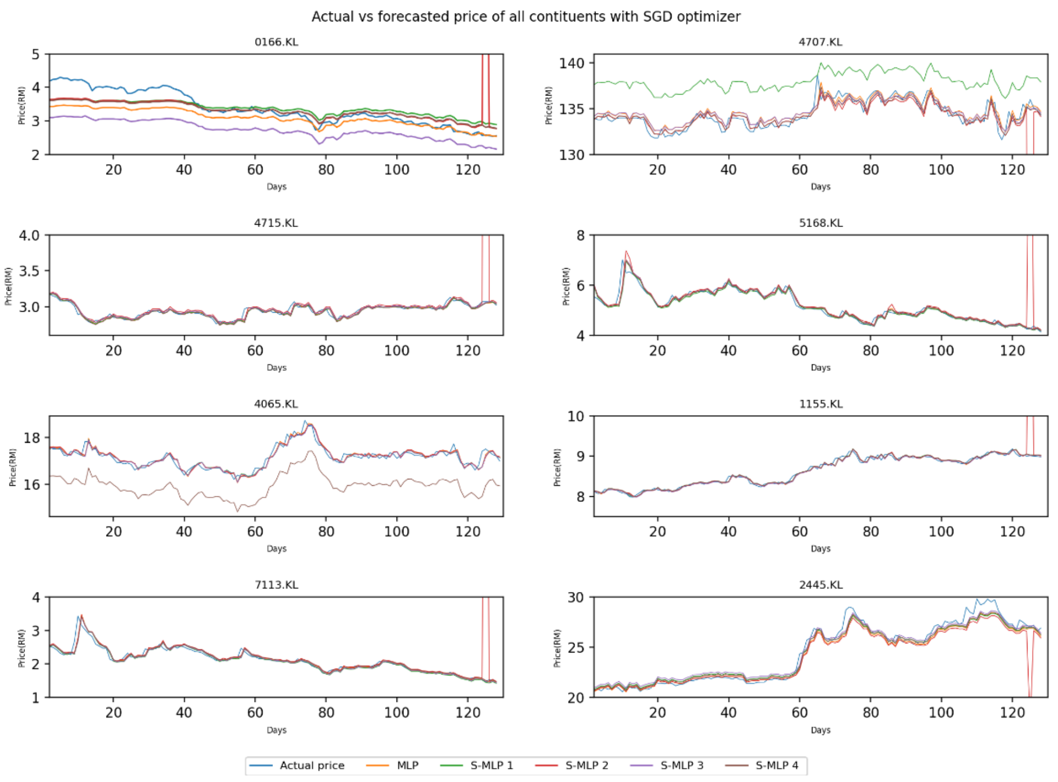

Figure 3. Comparison graph of actual versus forecasted stock price using all the neural networks with SGD optimizer

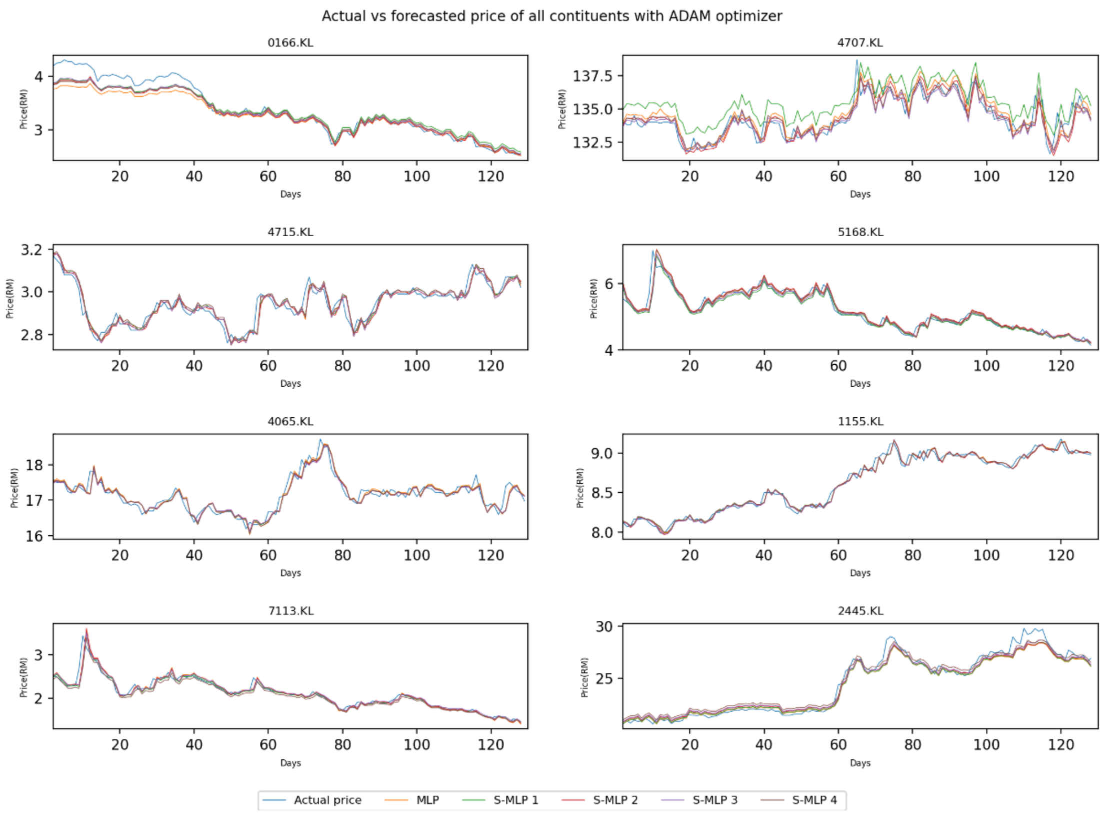

Figure 4. Comparison graph of actual versus forecasted stock price using all the neural networks with ADAM optimizer

Moving on, this research anticipated that S-MLP will perform better than MLP. As it can be observed from Figures 3 and 4 for most cases the forecasted values of MLP and variations of S-MLP are almost nearer to the actual value; except for 0166.KL, 4065.KL, and 4707.KL when it is trained with SGD and 4707.KL for neural network trained with ADAM. Hence, it cannot be definitively concluded that S-MLP is superior to MLP. Thus, further analysis is carried out to identify does S-MLP outperforms MLP by calculating the relative percentage improvement of the variations of S-MLP for all the constituents by letting the MLP as the reference model. The calculated relative percentage improvement is shown in Table 3.

Table 3. Relative percentage improvement for all the constituents with MLP as the reference model

| Optimizer | Model | Constituents | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0166.KL | 4707.KL | 4715.KL | 5168.KL | 4065.KL | 1155.KL | 7113.KL | 2445.KL |
| SGD | S-MLP 1 | -3.598 | 37.031 | -1.166 | -0.836 | -1.259 | 0.003 | -0.569 | -0.184 |
| | S-MLP 2 | -84.480 | 1.930 | -3.609 | -1.580 | -0.518 | -0.071 | -0.877 | 0.172 |
| | S-MLP 3 | -72.063 | -3.779 | -0.879 | -0.109 | -0.956 | -0.035 | -0.463 | 0.314 |
| | S-MLP 4 | -57.451 | 16.833 | -1.049 | -0.102 | -0.022 | -0.013 | -0.817 | 0.860 |
| ADAM | S-MLP 1 | 17.619 | -9.916 | -0.924 | -2.341 | -0.597 | -0.084 | -0.393 | 0.296 |
| | S-MLP 2 | 19.025 | 3.366 | -1.015 | -0.354 | -1.255 | -0.039 | -0.397 | 0.323 |
| | S-MLP 3 | 21.274 | -4.192 | -0.106 | -0.990 | -1.757 | -0.010 | -0.893 | 0.781 |
| | S-MLP 4 | 16.433 | -1.386 | -0.824 | -0.003 | -0.269 | -0.073 | -3.745 | 0.317 |

From Table 3 it can be observed that all the variations of S-MLP trained with ADAM outperformed for 0166.KL and 2445.KL only. Three variations of S-MLP with SGD have better accuracy than MLP for 4707.KL and 2445.KL constituents. MLP model has outperformed all the variations of S-MLP, for 4715.KL, 5168.KL, 4065.KL, and 7113.KL. This is because S-MLP were not able to escape the local minima or got stuck at the saddle point during minimization of loss function. However, the loss function in MLP was able to reach even lower local minima or reached the global minima during the learning process, therefore MLP outperformed S-MLP in most cases [26].

Additionally, an investigation is carried out to identify does the presence of stochastic sigmoid activation function during the testing times affects the accuracy of the forecasting. To carry out the investigation, a relative percentage improvement of S-MLP 3 and S-MLP 4 is calculated by letting S-MLP 1 and S-MLP 2 as the reference model respectively as shown in Table 4. Forecasting accuracy of five constituents (4715.KL, 5168.KL, 4065.KL, 7113.KL, and 2445.KL) of S-MLP 3 and S-MLP 4 with SGD optimizer exhibits an improvement. Furthermore, two constituents (4715.KL and 5168.KL) performed better than S-MLP 1 and S-MLP 2 when the ADAM optimizer is used to train the neural network. From Table 4, it can be observed and deduced that the presence of stochastic activation function during the prediction improves the S-MLP which only adapts the learning method proposed by [19].

Table 4. Relative percentage improvement for all the constituents with S-MLP without stochastic activation function during testing time as the reference model

| Optimizer | Model | Constituents | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0166.KL | 4707.KL | 4715.KL | 5168.KL | 4065.KL | 1155.KL | 7113.KL | 2445.KL |
| SGD | S-MLP 3 | -71.020 | -29.782 | 0.290 | 0.734 | 0.306 | -0.039 | 0.106 | 0.499 |
| | S-MLP 4 | 174.155 | 14.621 | 2.656 | 1.502 | 0.499 | 0.058 | 0.061 | 0.688 |
| ADAM | S-MLP 3 | 3.107 | 6.354 | 0.826 | 1.383 | -1.167 | 0.074 | -0.502 | 0.483 |
| | S-MLP 4 | -2.178 | -4.597 | 0.193 | 0.353 | 0.998 | -0.034 | -3.362 | -0.006 |

From this analysis, it can be concluded that in the big picture–the introduction of the stochastic sigmoid activation function into MLP can generate the pattern of the actual price when the stock price is forecasted. Moreover, the accuracy of the forecasting with S-MLP results in almost accurate forecasting. However, in most cases MLP outperformed S-MLP. Moving on, the presence of stochastic activation function in S-MLP during training and testing time performs better than the presence of stochastic activation function in S-MLP during training time only. Hence, it can be deduced that generally, MLP is the better model to forecast the stock price followed by S-MLP with sigmoid stochastic activation function during training time and testing time. Lastly, S-MLP with sigmoid stochastic activation function during training time only.

## 4. CONCLUSION

In the present manuscript, we introduced S-MLP by integrating stochastic sigmoid activation function into MLP which represents the noisy characteristic of the stock price. S-MLP mimics the stochastic environment of stock prices through the introduction of Gaussian noise and considered volatility estimators of the respective constituent into stochastic sigmoid activation function. The empirical analysis suggests that S-MLP did not outperform the deterministic model, MLP, except for three specific constituents: 0166.KL, 4707.KL, and 2445.KL. Besides that, the presence of stochastic sigmoid activation function in S-MLP during both the training and testing time achieved higher accuracy than S-MLP with stochastic activation function during training time only in most counters.

In future, it may be worth scaling the perturbation factor in stochastic sigmoid activation either by adding another parameter to the Gaussian process and making it as a hyperparameter to further improve the accuracy of forecasting. Apart from that, we suggest to further investigate why does all the variation of S-MLP performed better than MLP for 2445.KL and S-MLP with ADAM outperformed MLP for 0166.KL. Finally, we propose to extend this research on further examining why does the presence of stochastic activation function during test time performs better than S-MLP with sigmoid activation function during test time. We anticipate that the proposed S-MLP model will be able to contribute in a positive way to MLP research and the application of financial time series forecasting.

## REFERENCES

[1] C. L. Giles, S. Lawrence, and A. C. Tsoi, “Noisy Time Series Prediction using Recurrent Neural Networks and Grammatical Inference,” *Machine Learning*, vol. 44, no. 1, pp. 161–183, 2001, doi: 10.1023/A:1010884214864.

[2] Y. S. A. -Mostafa and A. F. Atiya, “Introduction to financial forecasting,” *Applied Intelligence*, vol. 6, no. 3, pp. 205–213, 1996, doi: 10.1007/BF00126626.

[3] M. Kumar and M. Thenmozhi, “Forecasting Stock Index Movement: A Comparison of Support Vector Machines and Random Forest,” in *Indian Institute of Capital Markets 9th Capital Markets Conference Paper*, 2006, pp. 1–16, doi: 10.2139/ssrn.876544.

[4] A. Namdari and T. S. Durrani, “A Multilayer Feedforward Perceptron Model in Neural Networks for Predicting Stock Market Short-term Trends,” *Operations Research Forum*, vol. 2, no. 3, pp. 1–30, 2021, doi: 10.1007/s43069-021-00071-2.

[5] B. R. Setyawati, R. C. Creese, and M. Jaraiedi, “Neural Networks for Univariate and Multivariate Time Series Forecasting Keywords,” in *IIE Annual Conference Proceeding*, 2003, pp. 1–6.

[6] M. Mallikarjuna and R. P. Rao, “Evaluation of forecasting methods from selected stock market returns,” *Financial Innovation*, vol. 5, no. 1, pp. 1–16, 2019, doi: 10.1186/s40854-019-0157-x.

[7] M. R. Islam and N. Nguyen, “Comparison of Financial Models for Stock Price Prediction,” *Journal of Risk and Financial Management*, vol. 13, no. 8, pp. 1–19, 2020, doi: 10.3390/jrfm13080181.

[8] D. L. Minh, A. S. -Niaraki, H. D. Huy, K. Min, and H. Moon, “Deep learning approach for short-term stock trends prediction based on two-stream gated recurrent unit network,” *IEEE Access*, vol. 6, pp. 55392–55404, 2018, doi: 10.1109/ACCESS.2018.2868970.

[9] M. M. Kumbure, C. Lohrmann, P. Luukka, and J. Porras, “Machine learning techniques and data for stock market forecasting: A literature review,” *Expert Systems with Applications*, vol. 197, pp. 1-41, Jul. 2022, doi: 10.1016/J.ESWA.2022.116659.

[10] E. Chong, C. Han, and F. C. Park, “Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies,” *Expert Systems with Applications*, vol. 83, pp. 187–205, 2017, doi: 10.1016/J.ESWA.2017.04.030.

[11] H. Ling, S. Samarasinghe, and D. Kulasiri, “Stochastic neural networks for modelling random processes from observed data,” *Studies in Computational Intelligence*, vol. 628, pp. 83–107, 2016, doi: 10.1007/978-3-319-28495-8_5.

[12] P. Jay, V. Kalariya, P. Parmar, S. Tanwar, N. Kumar, and M. Alazab, “Stochastic neural networks for cryptocurrency price prediction,” *IEEE Access*, vol. 8, pp. 82804–82818, 2020, doi: 10.1109/ACCESS.2020.2990659.

[13] J. Wang and J. Wang, “Forecasting stochastic neural network based on financial empirical mode decomposition,” *Neural Networks*, vol. 90, pp. 8–20, 2017, doi: 10.1016/J.NEUNET.2017.03.004.

[14] Z. Liao and J. Wang, “Forecasting model of global stock index by stochastic time effective neural network,” *Expert Systems with Applications*, vol. 37, no. 1, pp. 834–841, 2010, doi: 10.1016/J.ESWA.2009.05.086.

[15] J. Wang, H. Pan, and F. Liu, “Forecasting Crude Oil Price and Stock Price by Jump Stochastic Time Effective Neural Network Model,” *Journal of Applied Mathematics*, vol. 2012, pp. 1–15, 2012, doi: 10.1155/2012/646475.

[16] J. Wang, H. Pan, Y. Wang, and H. Niu, “Complex System Analysis on Voter Stochastic System and Jump Time Effective Neural Network of Stock Market,” *International Journal of Computational Intelligence Systems*, vol. 8, no. 4, pp. 787–795, 2015, doi: 10.1080/18756891.2015.1061397.

[17] J. Wang and W. Jun, “Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks,” *Neurocomputing*, vol. 156, pp. 68–78, 2015, doi: 10.1016/J.NEUCOM.2014.12.084.

[18] H. Mo and J. Wang, “Return scaling cross-correlation forecasting by stochastic time strength neural network in financial market dynamics,” *Soft Computing*, vol. 22, pp. 3097–3109, 2018, doi: 10.1007/s00500-017-2564-0.

[19] C. Gulcehre, M. Moczulski, M. Denil, and Y. Bengio, “Noisy Activation Functions,” in *Proceedings of The 33rd International Conference on Machine Learning*, 2016, pp. 3059–3068.

[20] L. C. G. Rogers and S. E. Satchell, “Estimating Variance From High, Low and Closing Prices,” *The Annals of Applied Probability*, vol. 1, no. 4, pp. 504–512, 1991, doi: 10.1214/aoap/1177005835.

[21] D. Yang and Q. Zhang, “Drift-independent volatility estimation based on high, low, open, and close prices,” *Journal of Business*, vol. 73, no. 3, pp. 477–491, 2000, doi: 10.1086/209650.

[22] D. Enke and S. Thawornwong, “The use of data mining and neural networks for forecasting stock market returns,” *Expert Systems with Applications*, vol. 29, no. 4, pp. 927–940, 2005, doi: 10.1016/J.ESWA.2005.06.024.

[23] M. I. C. Rachmatullah, J. Santoso, and K. Surendro, “A novel approach in determining neural networks architecture to classify data with large number of attributes,” *IEEE Access*, vol. 8, pp. 204728–204743, 2020, doi: 10.1109/ACCESS.2020.3036853.

[24] A. Shrestha and A. Mahmood, “Review of deep learning algorithms and architectures,” *IEEE Access*, vol. 7, pp. 53040–53065, 2019, doi: 10.1109/ACCESS.2019.2912200.

[25] H. -C. Shin, M. Orton, D. J. Collins, S. Doran, and M. O. Leach, “Organ Detection Using Deep Learning,” in *Medical Image Recognition, Segmentation and Parsing*, Massachusetts: Academic Press, 2016, pp. 123–153, doi: 10.1016/B978-0-12-802581-9.00007-X.

[26] B. Reddy and J. C. Usha, “Prediction of Stock Market using Stochastic Neural Networks,” *International Journal of Innovative Research in Computer Science & Technology*, vol. 7, no. 5, pp. 128–138, 2019, doi: 10.21276/IJIRCST.2019.7.5.1.
[27] F. Pedregosa *et al.*, “Scikit-learn: machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[28] D. Chicco, M. J. Warrens, and G. Jurman, “The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation,” *PeerJ Computer Science*, vol. 7, pp. 1–24, 2021, doi: 10.7717/peerj-cs.623.

## BIOGRAPHIES OF AUTHORS

**Assunta Malar Patrick Vincent** is currently pursuing a master’s degree with the Faculty of Ocean Engineering Technology and Informatics at Universiti Malaysia Terengganu. Her research field includes financial mathematics and machine learning. She can be contacted at email: p4445@pps.umt.edu.my.

**Hassilah Salleh** received her Ph.D. degree in stochastic analysis from the University of Oslo. She is currently a senior lecturer at Universiti Malaysia Terengganu. Her current research interests include stochastic analysis, financial mathematics, islamic financial mathematics, and machine learning. She can be contacted at email: hassilah@umt.edu.my.