Stock Price Forecasting Using Symbiotic Organisms Search Trained Neural Networks

Bradley J. Pillay[1] and Absalom E. Ezugwu[2(✉)]

[1] School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal, Westville Campus, Private Bag X54001, Durban 4000, South Africa
215031687@stu.ukzn.ac.za

[2] School of Computer Science, University of KwaZulu-Natal, King Edward Road, Pietermaritzburg Campus, Pietermaritzburg 3201, KwaZulu-Natal, South Africa
EzugwuA@ukzn.ac.za

Abstract. The prediction of stock prices is an important task in economics, investment and financial decision-making. This has for decades, spurred the interest of many researchers through their focused research in designing stock price predictive models. In this paper, the symbiotic organisms search algorithm, a new metaheuristic algorithm is employed as an efficient method for training feedforward neural networks. The training process is used to build a better stock price predictive model. The Straits Times Index, Russell 2,000, NASDAQ Composite and Dow Jones Industrial Average indices are utilized as time series dataset for training and testing the new system. Three evaluation methods namely, Root Mean Squared Error, Mean Absolute Percentage Error and Mean Absolute Deviation are used to compare the results of the implemented model. The results obtained revealed that the hybrid model exhibited outstanding predictive performance compared to the hybrid Particle Swarm Optimization, Genetic Algorithm, and Auto Regressive Integrated Moving Average based models. The new model is a promising predictive technique for solving high dimensional nonlinear time series data that are difficult to capture by traditional models.

Keywords: Stock price prediction · Symbiotic organisms search algorithm · Particle Swarm Optimization · Artificial Neural Networks

1 Introduction

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instruments traded on an exchange. The successful prediction of a stock's future price serves as a guide for investors to be cautious of their investments. The efficient-market hypothesis suggests that stock prices reflect all currently available information and any price changes that are not based on newly revealed information thus are inherently unpredictable [1]. Other schools of thoughts disagree and these individuals rely on various methods and technologies to gain insight on future price information [2].

© Springer Nature Switzerland AG 2019
S. Misra et al. (Eds.): ICCSA 2019, LNCS 11623, pp. 673–688, 2019.
https://doi.org/10.1007/978-3-030-24308-1_53

The value of a stock price is influenced by the earnings per share, firm's book value, price earnings ratio and dividends per share. Although these factors are the fundamental units influencing the base stock price, the market also reflects a power over a specific stock price at any point in time. This is due to the constant pull and push of demand and supply within the market, this fluctuation may be due to trader's personal preference, events portrayed by the news, strategic approaches to stock exchange or perceptions based on other traders behaviour. These fluctuations may be estimated based on past behavioural patterns of a particular stock to an extent, however random events that force the stock to behave out of its norm are very difficult to predict. These occurrences are what experienced traders look for in maximising their profits. As such, any insight on these anomalies prove to be highly valuable to any trader within the market [3].

Traditionally, stock price forecasting has been carried out using time series analysis [4]. With the emergence of Artificial Neural Networks (ANNs), this form of analysis could be effectively performed at scale with higher levels of accuracy and accountability for unconceived variables [5–7]. Furthermore, non-hybridized time series models are outperformed by ANNs and Auto Regressive Integrated Moving Average (ARIMA) hybrids [2]. Evaluation of the prediction accuracy of these approaches are undertaken through the computation of the Root Mean-Squared Error (RMSE) and the Mean Absolute Percentage Error (MAPE). However, this evaluation should represent how much financial value each possesses, as such performance is measured by profitability, consistency and robustness [8].

In this paper, an efficient hybrid symbiotic organisms search (SOS) algorithm, which is combined with a Feed-forward Neural Networks (FFNNs) is developed to solve the stock price prediction problem. The goals of this paper is therefore, to demonstrate the applicability of the SOS algorithm an ANNs and to also show that the new hybrid method is able to obtain better predications when compared with other existing methods that have been applied to solve the same problem. The ARIMA based model, Particle Swarm Optimization (PSO) algorithm, and Genetic Algorithm (GA), of which the last two are well-known global optimization metaheuristics are implemented and evaluated in order to demonstrate the outperformance of the proposed hybrid SOS algorithm.

The rest of the paper is organized as follows: Sect. 2 presents related work, which involves using FFNNs and hybrid models that consist of both metaheuristic algorithms. The motivation and methodology of the proposed prediction approach and three other implementation models are discussed in Sect. 3. Section 4 presents the experimental results using recent time-series datasets. Finally, the concluding remarks is given in Sect. 5.

2 Related Work

In the past decade, the application of artificial intelligence and machine learning techniques such the ANNs have been used for the forecasting of Straits Time Indices (STI). Several literatures tend to focus on portfolio optimisation which is the act of

selecting which stocks to invest your money in, given a finite capital and finite set of available stocks [9, 10]. This however is not the focus of this study, although it is a main focus point when considering utilising machine learning techniques for optimising return on investment.

In [8], the use of multiple methods for the prediction of stock prices was investigated. The authors used the stock prices of five different companies which were obtained from Yahoo Finance. The four different forecasting methods investigated were: ARIMA model, ANNs, Holt's Winters (a statistical forecasting method for seasonal time-series data) and time-series linear model (TSLM). It was found that the Holt's Winters method produced the best overall forecasting accuracy compared to the other methods [8]. In [11], the SOS algorithm was proposed for training a FFNNs. The computational results from the training with SOS were compared to other results obtained from similar training of FFNNs using other metaheuristic search algorithms, such as the culture search (CS), genetic algorithm (GA), particle swarm optimisation (PSO), mean-variance optimisation (MVO), gravitational search algorithm (GSA), and biogeography-based optimisation (BBO). The results showed that the SOS trained the FFNNs the best for the task at hand [11].

The study conducted in [12] used both the PSO and backpropagation algorithm to train a FFNNs for time-series forecasting. There were four types of time-series data used, these are sunspots (number of sunspots observed over a period of time), exchange-rate (the USD to INR exchange rate), earthquake (seismogram readings over time) and airline (the number of airline passengers). The results obtained were compared to results from other methods used to predict time-series data, such as the PSO-only trained FFNNs, backpropagation trained ANNs, and the Box-Jenkins models (which are statistically based models for predicting time-series data). Experimentation from the study showed that the PSO-only models were notably better than the backpropagation only models and that the hybrid approach (PSO + backpropagation) was better than the Box-Jenkins models [12].

The FFNNs variants proposed in [13] were used for stock market data (NAV of SBI mutual fund) prediction and evaluation of the performances of three different methods for adjusting the network weights during training: the resilient backpropagation method, the Levenberg-Marquardt (also referred to as Bayesian regularisation) method and the scaled conjugate gradient method. It was observed that the Bayesian regularisation method was the best at being able to generalise on the given data compared to the other training methods [13]. The study in [14] used the PSO algorithm to optimise the weights of an artificial neural network, which was used to forecast the exchange rate of the Straits Times Index (STI) time series data. The results obtained were very promising and interesting.

Therefore, building on the identified gap from the above related literature, the current study tries to replicate the earlier proposal made in [14] where the PSO algorithm was utilised for training neural networks. The current study, however, considers the employment of a more recent metaheuristic algorithm for the training of FFNNs with the main goal of building a more robust and efficient stock price predictive model.

In the next section, two of the main algorithms that inspired the current work, namely SOS and PSO are briefly discussed. Thereafter, the implementations of four hybrid models including SOS + FFNN, PSO + FFNN, GA + FFNN, and ARIMA model are explained.

3 Motivation and Methodology

The symbiotic organisms search algorithm is a new metaheuristic optimization algorithm that in recent time, attracted the attention of the research community because of the simplicity of its implementation and success records [15, 16]. The SOS algorithm has been widely used in many applications, such as parallel machine scheduling problem [17, 18], optimal allocation blood products [19, 20], and traveling salesman problem [21, 22]. The algorithm simulates the symbiotic interactions within a paired organism relationship that are used to search for the fittest organism [16]. The SOS iteratively uses a population of candidate solutions to promising areas in the search space in the process of seeking the optimal global solution. In the initial ecosystem, a group of organisms is generated randomly for the search space. Each organism represents one candidate solution and is associated with a certain fitness value, which reflects the degree of adaptation to the desired objective. The generation of new solutions is governed by three phases: the mutualism phase, commensalism phase, and parasitism phase. The nature of the interaction defines the main principle of each phase. Interactions benefit both sides in the mutualism phase; interactions benefit one side and do not impact the other in the commensalism phase; interactions benefit one side and actively harm the other in the parasitism phase. Each organism interacts with the other organism randomly through all phases. The process is repeated until termination criteria are met. The reader may refer to the work presented in [16] for an in-depth understanding of the fundamental design concept and computational representation of the three SOS global optimization search phases.

The SOS algorithm was chosen because of its successful implementations in related researches [16], the SOS algorithm is perceived to have the capability of yielding good results and performance when applied to stock price. Another reason for choosing SOS is that its operations require no specific control parameter. There are many advantages of SOS that also factored into the decision to consider this algorithm for the training of FFNNs. The algorithm avoids the risk of compromised performance due to improper parameter tuning. This is the case since the only parameters that need to be set are the size of the population/ecosystem and the maximum number of evaluations. Other algorithms such as the Genetic Algorithm (GA), Differential Evolution (DE), PSO, Mine Blast Algorithm (MBA), and Cuckoo Search (CS) require the tuning of at least more than one specific algorithm control parameters in addition to these two parameters. The SOS algorithm uses three interaction strategies, mutualism, commensalism, and parasitism, to gradually improve candidate solutions. This makes the algorithm simpler and quicker to implement since no time needs to be spent on the choice of operators. An organism (candidate solution) in this algorithm is represented by a vector

of size 2, where the values are the open and close stock values for a company. This representation was chosen since it is the most efficient way to represent all the necessary data and to be able to manipulate the data to get the best solutions.

The PSO algorithm used is the global best PSO hybridized with a neural network. In this algorithm, the neighborhood of each particle is the entire swarm. A swarm consists of a collection of particles, where each particle is a candidate solution. The particles are then evolved where each particle's position and velocity are changed according to its own experience and that of its neighbors. Each particle can communicate with every other particle, and each particle is attracted to the best particle found by any member in the swarm. Each particle is a point in an n-dimensional space and contains the set of all the weights in the neural network and the bias. The algorithm stops when the maximum number of iterations has been reached. The position of the ith particle is represented as $x_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ and these components of position represent the individual weights and bias. The velocity of the ith particle is represented as $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})$. There are no selection or evolutionary operators that are used. Instead, it uses a fitness function with updates of positions and velocities to find near optimal solutions.

The PSO algorithm has been chosen as a candidate competitive algorithm for the proposed SOS algorithm because it is a common algorithm used for stock price predictions. It is a good algorithm to compare SOS with since the results for the PSO implementation with neural networks has produced notable results for stock price prediction. It does not use operators such as mutation and crossover which makes it simpler and easier to implement. The search can be carried out by the speed of the particle. During the development of several generations, only the most optimist particle can transmit information onto the other particles, and the speed of the researching is very fast. The global best PSO model has been chosen since it converges faster than the I-best or the local best PSO models. This is due to the larger particle interconnectivity of the global best PSO model. However, global best PSO can easily be trapped in local minima, so more focus has to be given to exploration rather than exploitation during training. This is done by changing the PSO parameters such as higher values for the maximum velocity and inertia weight.

3.1 Modelling and Program Design

The model implementation for this study was coded in C# using Microsoft Visual Studio 2017 as the IDE. The program has a GUI interface where each of the three hybrid algorithms and ARIMA model can be run and the results displayed upon completion of the program run. The dataset used for training and testing contains data from 24 August 2017 to 1 November 2018 [24–27]. All the algorithms used the same dataset so that the performance comparisons between the algorithms would be more meaningful. All the algorithms were run on the same computer for 1000 iterations with a population size of 30. The details of the SOS, PSO, and GA with neural network algorithms and the ARIMA model are presented next.

3.2 Hybrid Symbiotic Organisms Search with Neural Networks

In order to improve the SOS algorithm, it is hybridized with a FFNNs. The idea of hybridizing SOS with a neural network was motivated by similar implementation method presented in [14], in which the PSO was hybridized with a neural network. Therefore, since hybridizing PSO with a neural network seemed to be a very common experience in the literature, it spurred interest on how SOS would perform if it was hybridized with a neural network. When it comes to stock price prediction there can be many companies involved and neural networks have good scalability to large datasets and work well with high dimensions. Neural networks also have the ability to model non-linear complex relationships and real-world stock market prediction is complex, so the application of neural networks will be beneficial. This hybridization works by using the SOS algorithm to train the neural network by finding the optimal weights and bias for the network in a similar way the PSO algorithm was used to train FFNNs in [14]. The network comprises of a single input layer with 2 nodes, a hidden layer with 8 nodes and an output layer with 2 nodes. A vector represents each organism or candidate solution, which contains the weights from the input layer to the hidden layer, the weights from the hidden layer to the output layer and the bias value for the network. This vector has a length of 34. The representation of the vector is illustrated as shown in Fig. 1.

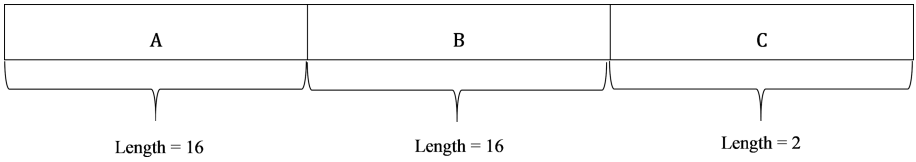

Fig. 1. Structure (Design) of candidate solution

A – weights linking the input layer nodes to the hidden layer nodes
B – weights linking the hidden layer nodes to the output layer nodes
C – the values for the bias.

The algorithm is trained using 240 instances which formed the training set. The testing set contains 60 instances since the dataset was split 80% for training and 20% for testing. The training and testing set was normalized independently after the train-test split. A population size of 30 is used and the algorithm is run over 1000 iterations. The algorithm takes two inputs: the open value and the close value for a stock and then predicts these two values for the next day. The RMSE is used as the fitness function in this algorithm since the goal is to minimize the error of the prediction, so an error formula is an appropriate fitness function.

Algorithm 1: Hybrid SOS with a Feed-forward Neural Networks

1 Initialize a population of size 30, composing of individuals described in Figure 1 (above). Each cell of the individuals is randomly initialized to values that are between 0 and 1.
2 REPEAT for each individual in the population
3 Initialize the weights of the neural network with the corresponding weights contained by the individual
4 REPEAT for each training instance
5 Input the instance in the FFNNs to obtain an output
6 Calculate RMSE for the output and the expected output
7 END
8 Calculate the Average of RMSE values which will serve as the fitness value of the individual
9 END
10 REPEAT
11 Increase number of iterations by 1
12 REPEAT for each individual X_i in the population
13 Set the best individual X_{best} to the individual with the lowest fitness value
14 MUTUALISM PHASE
15 Select an individual X_j randomly
Determine a mutual relationship vector
Mutual_Vector = $(X_i + X_j) / 2$
16 Determine the benefit factors BF1 and BF2, where the benefit factors are either 1 or 2
17 Modify X_{i_new} and X_{j_new} based on their mutual relationship
$X_{i_new} = X_i + rand(0,1) * (X_{best} - Mutual_Vector * BF1)$
$X_{j_new} = X_j + rand(0,1) * (X_{best} - Mutual_Vector * BF2)$
18 Calculate the fitness of X_{i_new} and X_{j_new} by using lines 3 to 8
19 IF X_{i_new} fitness value is less than X_i
20 Replace X_i with X_{i_new}
21 END
22 IF X_{j_new} fitness value is less than X_j
23 Replace X_j with X_{j_new}
24 END
25 END
26 COMMENSALISM PHASE
27 Select an individual X_j randomly
28 $X_{new} \leftarrow X_i + rand(-1,1) * (X_{best} - X_j)$
29 Calculate the fitness of X_{new}
30 IF X_{i_new} fitness value is less than X_i
31 Replace X_i with X_{new}
32 END
33 END
34 PARASITISM PHASE
35 Select an individual X_j randomly
36 Create a parasite ($X_{parasite}$) from X_i
37 Calculate the fitness of $X_{parasite-+}$
38 IF $X_{parasite}$ fitness value is less than X_j
39 Replace X_j with $X_{parasite}$
40 END
41 END
42 END
43 UNTIL number of iterations are equal to 1000

3.3 Hybrid Particle Swarm Optimization with Neural Networks

The second model implementation is the employment of the PSO hybridized with a FFNNs. The neural network consists of an input layer that has 2 nodes, a hidden layer that has 8 nodes and an output layer that has 2 nodes. The inputs are opening value for a stock and a closing value for a stock for a day. The network outputs the predicted opening and closing values for the stock for the next day. In this algorithm a swarm is initialized with 30 particles where each particle is represented by a vector of size 34 that holds all the weights for the network as well as the bias value. The swarm is also initialized with random velocities. The *minX* and *maxX* values are set to −1 and 1. The fitness function used is RMSE so that the error between the predicted values and the actual values can be minimized. The positions and velocities are updated for every iteration. The inertia value is set at 0.9, the two constants c_1 and c_2 are both set at 2 and the probability of death is 0.01. The algorithm runs until the maximum number of 1000 iterations are reached. The best positions after the PSO is run provides the optimal weights for the neural network to be able to predict the output values.

3.4 Hybrid Genetic Algorithm with Neural Networks

The third model implementation is the Genetic Algorithm hybridized with a FFNNs. The operators used includes the Roulette Selection, Uniform Crossover with a crossover rate set to 0.5, and Uniform Mutation with a mutation rate set to 1/(Length of individual) = 1/34. The population comprised of 30 individuals that were randomly initialized, and each individual is a vector of size 34. Elitism was used in selection for the population for the next generation, where the best individual is kept for the next generation. The fitness function used for this algorithm is the RMSE because we aim to minimize the error between the actual and predicted values. This algorithm was allowed to run for a maximum of 1000 iterations, and the best individual served as the optimal weights for the neural network to predict the output values.

3.5 Auto Regressive Integrated Moving Average

The fourth model is an Autoregressive Integrated Moving Average model. This implementation was done with an assistance of using the Extreme Optimization Numerical Libraries for .NET [23]. This library was built to assist developers to program financial, engineering and scientific applications. The auto regressive order was set to 0, the degree of differencing was set to 0 and the moving average was set to 2. These parameters yielded the best result compared to other combination of parameters settings tested on these datasets. Two ARIMA models were used, one to forecast the Open Stock Values for each day and the other was to forecast the Close Stock Values for each day.

3.6 Evaluation Metrics

A testing strategy that is used is the Mean Absolute Percentage Error (MAPE), which is a measure of the prediction accuracy of a forecasting method in statistics, for example

in trend estimation. It is a very common testing strategy for stock price prediction algorithms and many organizations focus primarily on MAPE when assessing forecast accuracy. Most people are also more comfortable when dealing with percentage terms which makes this error easy to interpret. The formula is given in Eq. 1.

$$MAPE = \frac{100\%}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right| \tag{1}$$

where A_t is the actual value of the stock price and F_t is the forecast value from the algorithm. The absolute value in this calculation is summed for every forecasted point in time and divided by the number of fitted points n. Multiplying by 100% makes it a percentage error. A drawback of this method is that it cannot be used for data that has zero values since this could result in a division be zero error. This model is used nonetheless because it is highly unlikely that the price of a stock will be zero. Due to the pitfalls in MAPE, it is used in conjunction with other evaluation techniques like the Mean Absolute Deviation (MAD). With MAPE the lower the percentage error the better.

Another common evaluation metric to test forecasting accuracy is the Root Mean Squared Error (RMSE). The RMSE is frequently used measure of the differences between values predicted by a model or an estimator and the values observed. This technique is used mainly when there is variance in the data, and it makes use of standard deviation which is good when it comes to mathematical operations. RMSE is the square root of the average of squared differences between prediction and actual observation. It expresses the average model prediction error in units of the variable of interest. The metric can range from 0 to infinity and is indifferent to the direction of error. The formula for RMSE is given in Eq. 2.

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_J)^2} \tag{2}$$

where n is the number of values, y_j is the forecast and the variable $\hat{y}_J$ is the mean error. Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means that the RMSE should be more useful when large errors are particularly undesirable. RMSE avoids the use of taking the absolute value, which is not wanted in many mathematical calculations. It is a negatively-oriented score, which means lower values are better.

The last testing metric discussed is the MAD. Other than MAPE, MAD is the most popular metric for evaluating forecast accuracy. The mean absolute deviation of a dataset is the average distance between each data point and the mean. This strategy measures variance just like MSE but lacks the strong statistical relationship MSE has. MAD has the advantage of being easier to understand among people who are not

specialists in the field, and this is partly due to the fact that the error has the same dimension as the forecast. The formula for calculating MAD is represented in Eq. 3.

$$MAD = \frac{1}{T}\sum_{t=1}^{T}|e_t - \hat{e}_t| \tag{3}$$

where T is the number of time periods, e_t is the forecast error in period t and the last term denoted by $\hat{e}_t$ is the mean error for period t. The metric MAD is used in conjunction with MAPE to help overcome the pitfalls of MAPE and give a better overall view of the results. The three testing metrics are used to ensure that the error of the forecast can be seen using different evaluations to make it easier to determine which forecasting algorithm produces the best results. This combined evaluation technique allows for a better comparison between the algorithms, so that a more informed decision can be made.

4 Experimental Results

All the algorithms were run on the same Micro-Star International (MSI) computer to allow for better comparison of the results. The computer specifications are indicated as follows: Processor: Intel® Core™ i7-7700HQ, CPU @ 2.80 GHz, Installed Memory (RAM): 12.0 GB, GPU: Nvidia GTX1050, System Type: 64-bit Operating System, x64-based processor Operating System: Windows 10 Home. Each algorithm was run 20 times and the average of the results was recorded. All the simulation results presented in this paper was generated using the stock price prediction GUI simulator shown in Figs. 2 and 3.

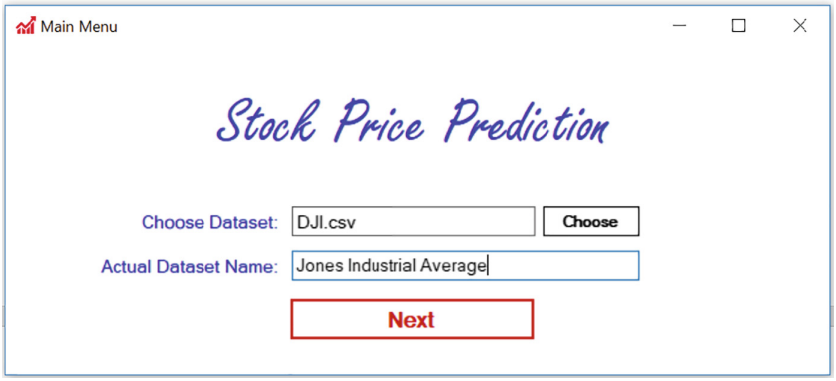

Fig. 2. Stock price prediction user interface with datasets file selection phase



Fig. 3. Stock price prediction user interface with algorithm selection phase

4.1 Results and Discussion

The dataset used in this research are the Straits Times Index [24], NASDAQ Composite [25], Russel 2000 [26], and Dow Jones Industrial Average [27] financial stock data. The three hybrid algorithms were executed 20 times for each dataset and compared to the ARIMA model using parameters specified in Sect. 3.5. The results for each execution were evaluated using RMSE, MAD and MAPE. These evaluation metrics was applied to the open and close stock values independently. Thereafter, the final RMSE, MAD and MAPE values were calculated by taking the average evaluations obtained for the open and close stock values for each execution. Table 1 below displays the average results obtained by making use of the Straits Times Index (STI) dataset [24]. The algorithms that predicts values that are higher than the actual values obtain MAPE scores that are higher than 100%. All algorithms except the hybrid SOS model obtained MAPE values are significantly greater than 100%, as shown in Table 1 below. The ARIMA model obtained the highest average MAPE value. The hybridized PSO obtained the second highest average MAPE value, while the hybrid GA obtained the third highest average MAPE value. The SOS hybridized model achieved the lowest average MAPE value, together with the lowest MAD and RMSE values. All the MAD and RMSE values obtained by the four algorithms are close to zero indicating a small prediction error.

Table 1. Average results of different algorithms executed on Straits Times Index (STI) dataset

	Average RMSE	Average MAPE	Average MAD
ARIMA	0.27061	602.45719	0.22998
GA + FFNN	0.158544925	247.2920749	0.136108794
PSO + FFNN	0.178063764	287.124298	0.148943577
SOS + FFNN	**0.117305261**	**95.01212301**	**0.094189188**

Table 2 displays the average results obtained by making use of the Dow Jones Industrial Average (DJI) dataset [24]. For this dataset, all the algorithms obtained MAPE values extremely greater than 100%. The hybrid SOS model received the lowest average MAPE value, with hybrid PSO model following it. The hybrid GA obtained the third lowest average MAPE value and ARIMA achieved the highest MAPE value. All the algorithms received RMSE and MAD values that are very close to zero, hence, there are small prediction errors. The MAPE value obtained by the hybrid SOS model is more than half of the second lowest MAPE value which was achieved by the PSO hybridized model.

Table 2. Average results of different algorithms executed on the Dow Jones Industrial Average (DJI) dataset

	Average RMSE	Average MAPE	Average MAD
ARIMA	0.29781	4141.64874	0.24341
GA + FFNN	0.169998799	1958.530227	0.141143727
PSO + FFNN	0.191264126	1704.068111	0.157681562
SOS + FFNN	**0.144319287**	**841.839545**	**0.118860402**

Table 3 below displays the average results obtained by making use of the NASDAQ Composite dataset [25]. All four algorithms received MAPE values greater than 100%. However, the ARIMA model produced an average MAPE value that is the highest. The GA hybridized algorithm obtained the second highest MAPE value and PSO hybridized received the third highest. SOS hybridized proved again to obtain the lowest MAPE value and receiving MAD and RMSE values that are the closest to zero. The other algorithms also achieved MAD and RMSE values close to zero, showing a low prediction error, as shown in Table 3.

Table 3. Average results of different algorithms executed on the NASDAQ Composite dataset

	Average RMSE	Average MAPE	Average MAD
ARIMA	0.28175	2195.20135	0.23301
GA + FFNN	0.171553204	1654.548538	0.146370212
PSO + FFNN	0.168955706	933.3688597	0.140076609
SOS + FFNN	**0.123727032**	**785.8357967**	**0.102633434**

Table 4 displays the average results obtained by making use of the Russel 2000 (RUT) dataset [26]. All algorithms obtained MAPE values higher than 100%. The SOS hybridized obtained the lowest average MAPE, RMSE and MAD values. The highest MAPE value was obtained by ARIMA followed by hybrid GA and tagged behind is the hybrid PSO algorithm. However, the RMSE and MAD values follow a different trend. The highest RMSE and MAD values was obtained by ARIMA followed by the hybrid PSO and then the hybrid GA. The hybrid SOS algorithm obtained values that are small and close to zero.

Table 4. Average results of different algorithms executed on the Russel 2000 dataset

	Average RMSE	Average MAPE	Average MAD
ARIMA	0.30676	832.86160	0.27815
GA + FFNN	0.169451308	459.2586704	0.145671891
PSO + FFNN	0.171527905	359.3365194	0.144370396
SOS + FFNN	**0.110566035**	**237.4466027**	**0.090007681**

Overall, it can clearly be seen that the hybrid SOS with a FFNNs outperforms the hybrid PSO, hybrid GA and ARIMA model for all the three evaluation techniques. Analyzing the results closely it can be seen that the hybrid SOS has lower scores for both RMSE and MAD values than the other three algorithms, which means that there is less variability in the data. Hence, the results are much more clustered around a line of best fit. Therefore, the findings of the current study reveals that the proposed hybrid SOS can be used as an alternative robust stock prices prediction algorithm.

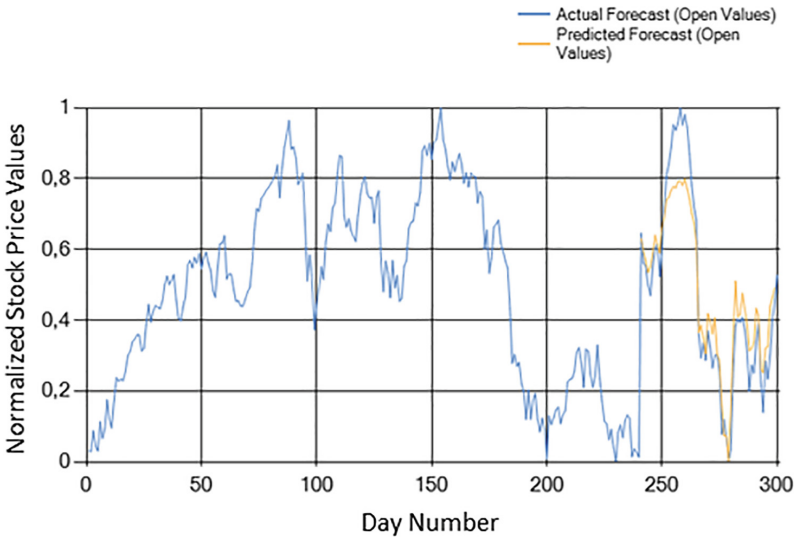

Fig. 4. STI actual and predicted open stock market values forecast using the SOS hybrid model (Color figure online)

Figure 4 above illustrates the forecast for the open stock values, while Fig. 5 shows the forecast for the close stock values for an execution of the SOS hybrid model on the STI dataset. The horizontal axis represents the days in chronological order, and the vertical axis represents the normalized stock values. The blue line graph in figure Figs. 4 and 5 depict the actual forecast for the given time period. Whilst, the orange line is the predicted forecast which starts around the 240th day mark. Comparing the actual and predicted forecast we can see that the predicted forecast is able to follow the same trend as the actual forecast. Majority of the predictions are really close to the

actual values. However, when the actual stock values are greater than 0.8 for the open stock value forecast graphs, the predicted stock values do not come as close to the actual stock values, but the underestimation is a difference approximately below 0.2.

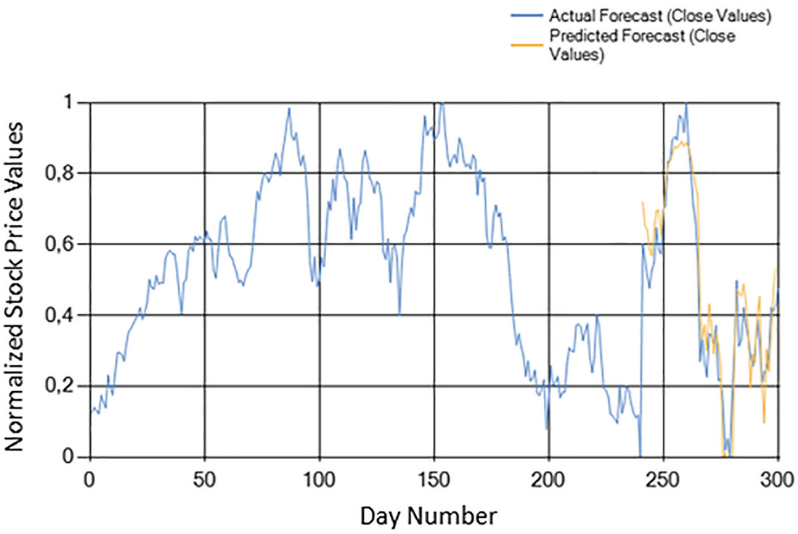

Fig. 5. STI actual and predicted close stock market values forecast using the SOS hybrid model (Color figure online)

In summary, this study was conducted based on an existing stock price-forecasting model that was implemented using PSO-trained neural networks. This model was initially trained on STI data of a greater magnitude, this information has since been removed from the source and access to a much smaller data set could be acquired. As such the model for the current study was trained using the same data of smaller magnitude with similar weighting pertaining to training and testing. This model was successfully replicated. Afterwards the SOS algorithm was chosen to replace the PSO component of the model with the hope of achieving greater optimization efficiency. SOS excels in many areas with similar attributes to PSO, as such SOS seemed viable as a substitute. The implementation was modelled without a FFNNs to identify the relationship between the algorithm and the problem without overlapping influence from the FFNNs. This model continued to show promise as the results experienced surpassed that of the PSO, GA, algorithms with a FFNNs. This inspired a further adaptation of the model to include a FFNNs to not only gain insight on this relationship but also in hopes of achieving better results.

5 Conclusion

The SOS with FFNNs model was developed and tested based on the existing combination of PSO and FFNNs study. Additional comparison that involves the implementation of hybrid GA with FFNNs and ARIMA model were carried out to further

validate the superior performance of the hybrid SOS algorithm. The hybrid SOS with FFNNs outperformed hybrid PSO, GA with FFNNs and ARIMA models by noticeable margins. The shortcomings of SOS with FFNNs has been identified and attributed to the increased implementation complexity given by the combination of two already complex algorithms. Future improvements of SOS with FFNNs could include training on a much larger data set or data sets with much higher complexity levels. This could then be adapted to incorporate multi-objective parameters between relative stock prices that may influence another stock's price. Finally, a model consisting of a FFNNs with a hybridization that utilizes the SOS optimization algorithm shows promise in the area of stock price prediction and supersedes that of the PSO, GA, ARIMA models implementation. However, the added complexity of a FFNNs may prove to be an area that requires greater fine-tuning to achieve a better predictive accuracy.

References

1. Pownall, G., Wasley, C., Waymire, G.: The stock price effects of alternative types of management earnings forecasts. Acc. Rev. **68**, 896–912 (1993)
2. Pai, P.-F., Lin, C.-S.: A hybrid ARIMA and support vector machines model in stock price forecasting. Omega **33**(6), 497–505 (2005)
3. Seetharaman, A., Niranjan, I., Patwa, N., Kejriwal, A.: A study of the factors affecting the choice of investment portfolio by individual investors in Singapore. Acc. Financ. Res. **6**(3), 153 (2017)
4. Montgomery, D.C., Johnson, L.A., Gardiner, J.S.: Forecasting and Time Series Analysis. McGraw-Hill, New York etc. (1990)
5. Refenes, A.N., Zapranis, A., Francis, G.: Stock performance modeling using neural networks: a comparative study with regression models. Neural Netw. **7**(2), 375–388 (1994)
6. Schöneburg, E.: Stock price prediction using neural networks: a project report. Neurocomputing **2**(1), 17–27 (1990)
7. White, H.: Economic Prediction using Neural Networks: The Case of IBM Daily Stock Returns, pp. 451–458. University of California, Oakland (1988)
8. Ponnam, L.T., Rao, V.S., Srinivas, K., Raavi, V.: A comparative study on techniques used for prediction of stock market. In: International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT), pp. 1–6. IEEE (2016)
9. Konno, H., Yamazaki, H.: Mean-absolute deviation portfolio optimization model and its applications to Tokyo stock market. Manag. Sci. **37**(5), 519–531 (1991)
10. Trippi, R.R., By-Lee, P., Jae, K.: Artificial Intelligence in Finance and Investing: State-of-the-Art Technologies for Securities Selection and Portfolio Management. McGraw-Hill Inc., New York (1995)
11. Wu, H., Zhou, Y., Luo, Q., Basset, M.A.: Training feedforward neural networks using symbiotic organisms search algorithm. Comput. Intell. Neurosci. **2016**, 14 (2016)
12. Adhikari, R., Agrawal, R.K.: Hybridization of artificial neural network and Particle Swarm Optimization methods for time series forecasting. Int. J. Appl. Evol. Comput. (IJAEC) **4**(3), 75–90 (2013)
13. Jabin, S.: Stock market prediction using feed-forward artificial neural network. Int. J. Comput. Appl. **99**(9), 4–8 (2014)
14. Junyou, B.: Stock price forecasting using PSO-trained neural networks. In: IEEE Congress on Evolutionary Computation 2007, CEC 2007, pp. 2879–2885. IEEE (2007)

15. Ezugwu, A.E., Adeleke, O.J., Akinyelu, A.A., Viriri, S.: A conceptual comparison of several metaheuristic algorithms on continuous optimisation problems. Neural Comput. Appl. 1–45 (2019)
16. Ezugwu, A.E., Prayogo, D.: Symbiotic organisms search algorithm: theory, recent advances and applications. Expert Syst. Appl. **119**, 184–209 (2019)
17. Ezugwu, A.E., Adeleke, O.J., Viriri, S.: Symbiotic organisms search algorithm for the unrelated parallel machines scheduling with sequence-dependent setup times. PLoS ONE **13** (7), e0200030 (2018). https://doi.org/10.1371/journal.pone.0200030
18. Ezugwu, A.E.: Enhanced symbiotic organisms search algorithm for unrelated parallel machines manufacturing scheduling with setup times. Knowl.-Based Syst. **172**, 15–32 (2019)
19. Govender, P., Ezugwu, A.E.: A symbiotic organisms search algorithm for optimal allocation of blood products. IEEE Access **7**, 2567–2588 (2019)
20. Govender, P., Ezugwu, A.E.: A symbiotic organisms search algorithm for blood assignment problem. In: Blesa Aguilera, M.J., Blum, C., Gambini Santos, H., Pinacho-Davidson, P., Godoy del Campo, J. (eds.) HM 2019. LNCS, vol. 11299, pp. 200–208. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-05983-5_16
21. Ezugwu, A.E., Adewumi, A.O., Frîncu, M.E.: Simulated annealing based symbiotic organisms search optimization algorithm for traveling salesman problem. Expert Syst. Appl. **77**, 189–210 (2017)
22. Ezugwu, A.E., Adewumi, A.O.: Discrete symbiotic organisms search algorithm for travelling salesman problem. Expert Syst. Appl. **87**, 70–78 (2017)
23. https://www.extremeoptimization.com/Default.aspx. Accessed 12 Dec 2018
24. https://finance.yahoo.com/quote/%5ESTI%3FP%3D%5ESTI/history/. Accessed 02 Dec 2018
25. NASDAQ Composite dataset. https://finance.yahoo.com/quote/%5EIXIC/history?period1=1511987617&period2=1543523617&interval=1d&filter=history&frequency=1d. Accessed 29 Nov 2018
26. Russel 2000 (RUT) dataset. https://finance.yahoo.com/quote/%5ERUT/history?period1=1511987909&period2=1543523909&interval=1d&filter=history&frequency=1d. Accessed 29 Nov 2018
27. Dow Jones Industrial Average. https://finance.yahoo.com/quote/%5EDJI/history?period1=1511987909&period2=1543523909&interval=1d&filter=history&frequency=1d. Accessed 29 Nov 2018