I.J. Intelligent Systems and Applications, **2020, 5, 52-72**
Published Online October 2020 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2020.05.05

MCS-MCMC for Optimising Architectures and Weights of Higher Order Neural Networks

Noor Aida Husaini
Faculty of Computer Science & Information Technology, Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Batu Pahat, Johore, Malaysia
E-mail: noor.aida.husaini@gmail.com

Rozaida Ghazali, Nureize Arbaiy and Ayodele Lasisi
Faculty of Computer Science & Information Technology, Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Batu Pahat, Johore, Malaysia
E-mail: {rozaida, nureize}@uthm.edu.my, lasisiayodele@yahoo.com

Received: 17 October 2019; Revised: 20 January 2020; Accepted: 16 March 2020; Published: 08 October 2020

Abstract: The standard method to train the Higher Order Neural Networks (HONN) is the well-known Backpropagation (BP) algorithm. Yet, the current BP algorithm has several limitations including easily stuck into local minima, particularly when dealing with highly non-linear problems and utilise computationally intensive training algorithms. The current BP algorithm is also relying heavily on the initial weight values and other parameters picked. Therefore, in an attempt to overcome the BP drawbacks, we investigate a method called Modified Cuckoo Search-Markov chain Monté Carlo for optimising the weights in HONN and boost the learning process. This method, which lies in the Swarm Intelligence area, is notably successful in optimisation task. We compared the performance with several HONN-based network models and standard Multilayer Perceptron on four (4) time series datasets: Temperature, Ozone, Gold Close Price and Bitcoin Closing Price from various repositories. Simulation results indicate that this swarm-based algorithm outperformed or at least at par with the network models with current BP algorithm in terms of lower error rate.

Index Terms: Modified Cuckoo Search, Markov chain Monté Carlo, MCS-MCMC, Higher Order Neural Network, weight optimisation, Backpropagation.

1. Introduction

Neural Network (NN) is a sequence of algorithms that attempt to identify the underlying relationships in a data set by means of a mechanism which imitates how the human brain works. It has more outstanding learning skills and is commonly used for more complex tasks such as mastering handwriting, identification of languages and prediction. There are different types of NNs like feedforward networks, Recurrent Neural Networks (RNN), Higher Order Neural Networks (HONN), etc. Among these types, HONN like Functional Link Neural Network (FLNN) and Pi-Sigma Neural Network (PSNN) can cope with high frequency, non-linear, discontinuous data and capable of providing explanations for their behaviour ("open boxes") [1]. They are robust and solve issues such as poor training rate, premature convergence faced in conventional NN. Instead, HONN's computing, processing and learning capabilities are remarkable. This is because, the HONN's structure that can be tailored to the structure of a problem [1].

Nevertheless, it is a very important job to initialise number of layers and weight when designing an NN. Therefore, it is vital to optimise the parameter in the HONN to increase the performance rate. This is because, during training, in order to minimise the error function, the weights of the linear combination need to be adjusted so that the desired network input-output relationship can be achieved. Some methods can be used to change weights in NN, i.e., Backpropagation (BP), Genetic Algorithm (GA) [2,3], Evolutionary Algorithm (EA) [4,5] and so on. NN's weight optimisation phenomenon has become an active area of research. Needless to say, Swarm Intelligence (SI) has also played a part. Among these swarm-based algorithms that have gained massive success in the past few years are GA [2,3], EA [4,5], Differential Evolution (DE) [6,7], and Artificial Bee Colony [8].

Not to forget, Whale Optimisation Algorithm [9] has been used to optimise the weights and biases. Based on the findings, this algorithm has demonstrated the ability to solve a wide range of optimisation problems and outperform the current BP learning algorithm. Another work is using DE to change the weight parameters encoded within the structure. In short, they used GA is to optimise the network topology and DE to set the network weights [10]. In another study, Salimans and Kingma optimised the weight to speed up the convergence rate by reparameterising the weight vectors in an NN [11]. In particular, the main reason for optimising weight parameters is to avoid local minima and convergence speed. This is due to the weights are the relative strength of the connections between nodes after the NN training that contains knowledge. These swarm-based algorithms were chosen due to the use of a common randomisation arbitration and local search to minimise or optimise such objective functions or error functions.

The Cuckoo Search (CS), which lies within this area of swarm-based algorithms, is based on the obligate brood parasitic behaviour of some cuckoo species in combination with the Lévy flight behaviour of some birds and fruit flies [12]. The CS algorithm uses high-level techniques for search space exploration and exploitation in which, in the long run, the step-length is much shorter. Since its discovery, the CS algorithm has been used extensively with positive results [13-16]. Because of less parameter settings and rate of convergence, the CS algorithm was examined in the sense of simplicity, making the algorithm more versatile and robust for many optimisation problems. Various modification related to CS algorithm has been made. For instance, by adjusting the parameters, P_α and α , Valian et al. [17] introduced an Improved Cuckoo Search algorithm. Such parameters are important if the solution vectors are to be finely tuned. For the time being, Walton et al. [18] changed the CS algorithm by incorporating the additional step of information exchange between the top eggs to accelerate convergence and reduces the computational cost. Therefore, this research work intent to integrate the ability of SI with NN thus that the benefits from both methods can be utilised to recover each drawback.

Motivated by the condition as mentioned earlier, this research used the Modified Cuckoo Search (MCS) with Markov chain Monté Carlo (MCMC) for training and out-of-sample observation for the predictive task. On this point, the weights optimisation in HONN’s models; namely FLNN and PSNN has been modelled using optimisation technique called Modified Cuckoo Search-Markov chain Monté Carlo (MCS-MCMC) algorithm [19]. Those hybridisations could assist in exploiting and uniting the advantages of each individual pure strategy for constructing better performances.

The performance of MCS-MCMC has been tested with several available repositories. Therefore, this research is carried out to prove that MCS-MCMC is ideal for updating the trainable weights in HONN and optimising the accuracy rate. Be mentioned, even different modifications and hybridisation being proposed, still, there is remaining significant gaps in this particular line of research. Often, such combination has empirically been overshadowed in practice by using the simple average instead. Therefore, it is not easy to extend these operators to solve different optimisation problems. In fact, research in SI involves high-level analysis about the performances. To meet this challenge, the needs for detailed analysis to determine the appropriate parameter measurement thus the hybridisation works better in order to find optimal solutions is still questioned.

2. Optimisations

2.1. *Previous Techniques in Optimisations*

Optimisations are made in a wide range of diverse disciplines [13,18,20,21]. A domain of optimisation problem has strong non-linearity in general. So far in the literature, three (3) main approaches are generally used for optimisation: 1) Conventional numerical optimisation methods, 2) stochastic optimisation methods and 3) hybrid methods. Some specific applications to this optimisation problem include the work of Walton et al. [18] in solving structural engineering problems. Further, Zolesio’s speed method was extended to structural optimisation and sensitivity analysis by Dems and Mroz [22] . On the other hand, an essential category of domains was introduced by Chenasis [23]. The results showed the presence in the weak sense of the ideal field. On the other hand, Fujii [24] deals with second order necessary conditions for an optimisation problem with a Dirichlet problem based on the conditions needed for the second order. He found that the second order is necessary for an optimisation problem with the assumption of differentiability of a Neumann problem. In addition, he also spoke about the existence of the optimal domain in the classical sense by applying a topology to a domain category [25]. At the same time, Fujii [26] had also proved that objective functional’s lower semicontinuity properties depend on the solution gradient to a limit value problem. However, those approaches which mostly based on mathematical approaches are often inefficient and adopts computationally training algorithm. The reason is that, they do not complete the requirements of robustness in big or highly dimensional spaces. Thus, it is somehow a difficult task to find the optimal values.

Despite, various NN were applied in optimisation with varying degrees of success. For instance, Zheng [27] hybrids Fireworks Algorithm (FA) with DE focusing on DE exploration and FA exploitation. Combining the methods proved to be the favoured strategy, rather than using individual models. Gülcü [28] combined Ant Colony Optimisation (ACO) algorithm, Particle Swarm Optimisation (PSO) algorithm and 3-Opt algorithm. The PSO algorithm is used for optimising the parameter values which are used for city selection operations in the ACO algorithm and determines significance of inter-city pheromone and distances. 3-Opt heuristic method is added to the proposed method in order to improve the local solutions. The performance of the combined method becomes very significant in terms of solution quality and robustness. Arabasadi et al. [29] combines GA and NN with the purpose of increasing the performance of NN and getting a high accuracy rate in diagnosing coronary artery disease. This somewhat shows surprising results that make those combination achieved accuracy, sensitivity and specificity rates.

Taking into account, Lahmiri [30] combined multiresolution analysis techniques and NN for predicting the next-day variation prediction. He adopted PSO to optimise its initial weights. The findings showed comprising results as well as providing good forecasting performance. Antipov et al. [31] optimised the "Identity-Preserving" by emphasising the original person's identity in the aged version of his/her face using synthetic images of exceptional visual fidelity, generated by the proposed Generative Adversarial Networks. On the other hand, the findings of Springenberg et al. [32] demonstrated that Bayesian optimisation can be used to optimise the hyperparameters and learning method in Deep Neural Networks. Wang et al. [33] suggested the hybridisation of Krill herd (KH) and quantum-behaved PSO (QPSO), called KH-QPSO for enhancing the ability of the local search and increasing the individual diversity in the population. The results significantly proved that the KH-QPSO is capable of avoiding the premature convergence and eventually finding the function minimum. Thus, making all the individuals proceed to the true global optimum without introducing additional operators to the basic KH and QPSO algorithms. On the other hand, Jin and Jin [34] considered hybridising different methods using NN and QPSO for software fault-prone prediction. Hence, combining both approaches in which the NN operates to classify the software modules into fault-proneness or non-fault-proneness categories while QPSO reduced the dimensionality.

Given the shortcomings of the conventional based techniques and the numerical ones, the SI methods have increased their popularity including CS [6,14,15,35]. The optimal solutions obtained by the CS are far better than the best solutions obtained by efficient particle swarm optimisers [36] and GA [15]. They have provided better solutions to most complex real-world optimisation problems compared to conventional numerical methods.

2.2. Issues in Optimisations

To note, there is no universal optimisation algorithm except for a series of algorithms, each customised to a problem of optimisation. Optimisation algorithms were called upon to address much larger and more complex problems than in the past. Consequently, it is performed to discover the best of the many alternatives available with the most cost effective and highest achievable performance. An optimisation problem requires a set of decision variables to be optimised. To ensure certain objective can be achieved, keeping in view certain constraints to be satisfied by the optimised solution. When dealing with optimisations, one must confront the nonconvex optimisation problems. The nonconvex case has multiple locally optimal points in which those cases may take a lot of times or forever to be solved. In such a scenario, the objective functions and constraints must be carefully designed thereby convex optimisation can be achieved.

In conjunction with that, the optimisation treats important topics such as having a way to exploit and explore the search space of the problem and includes comprehensive discussion of handling those continuous domains [37,38]. Therefore, it is a necessary to control the way of manipulating these two (2) issues. The challenge is to choose the right methods of optimisation, as it can decide whether the problem solves quickly or slowly. Undeniably, for achieving the optimal solutions. Most of the optimisation approach has been theoretically proved [37]. Therefore, more experiments should be carried out in order to prove those theoretical foundations. For instance, the mathematical analysis can be used to help in detailing the behaviour of the algorithms.

Instead, one of the well-known solutions is to consider the variables' values that maximise the targets. In some cases, the variables are often limited or excluded. In order to identify those values, experiments should focus on component-view parameter, supplementary to those experiments. The parameter needs to be optimised subsequently to build up such appropriate and effective models. Once the effective models being developed, then the parameter is in its optimality conditions. The practice of optimisation depends not only on how efficient and robust the algorithms are, as well as on good modelling techniques and careful results interpretation. However, there is an issue rises on how to construct an appropriate model, which is the important step in the optimisation process. If the model is too simplistic, the practical problem will not be presented with useful insights. Otherwise, it may be too difficult to solve if it is too complicated. Therefore, by expressing those uncertainties in regions with little or no data, it may lead those uncertainties to more systematic exploration.

3. Swarm-Based Learning Algorithm

To address the shortcomings of the traditional gradient-based algorithm in FLNN and PSNN, this research aims to improve the network learning algorithm by using swarm-based optimisation techniques. This swarm-based optimisation technique aims to define the parameter values which will determine the objective function that is desired to get the maximum or minimum value. With the objective function, these techniques are trying to arrive at a target for output by looking up at the relationships between optimisation parameters such as the objective function and the optimisation criterion.

In substance, both FLNN and PSNN's structures are converted into an objective function. The error criterion is set for both networks as the optimisation criterion. Thenceforth, the training procedure is then performed to minimise the error criterion. This can be done by tuning the weights of both FLNN and PSNN in the objective function. During the training process, adjustments of FLNN and PSNN weights parameters are based on swarm intelligence optimisation techniques for local search and global search strategies. The network error for both FLNN and PSNN is used as a fitness metric to direct the simulations towards optimal solutions. Those repeated processes of searching are stopped until the stopping criteria met. As in Fig.1 the steps involved in the learning process for FLNN and PSNN are outlined.

```
Step 1:    GENERATE population of solutions (weight values of
           FLNN and PSNN).
Step 2:    UPDATE FLNN and PSNN weight based on learning
           algorithm procedure.
Step 3:    EVALUATE fitness values (FLNN and PSNN network
           error) to select optimal solution.
Step 4:    IF the desired result is obtained, THEN stop.
Step 5:    ELSE, repeat Step 1.
```

Fig.1. Pseudocode for FLNN and PSNN training procedure

3.1. *MCS-MCMC Algorithm for FLNN (FLNN-MCMC)*

In this section, the use of MCS-MCMC to train the FLNN network (FLNN-MCMC) is discussed. The FLNN-MCMC algorithm used standard FLNN architecture without any changes. The changes only involving the replacement of current BP algorithm with MCS-MCMC. The replacement is made to resolve the current BP algorithm's gradient-based drawbacks. The general idea of implementing the MCS-MCMC in FLNN-MCMC is to search the optimal weights values of standard FLNN. The algorithm is used for weight initialisation and weight update (replacing the current BP algorithm in the standard FLNN). The weights and biases were calculated and updated for the complete training that represents the architecture. Those can be achieved by starting it with random values followed by frequent attempts to find better solutions and abandon the poor values. The architecture of FLNN-MCMC is presented in Fig.2. x_i, x_j, x_k is the input vector, w_{ijk} is the adjustable weight, y is the output and σ is the non-linear activation function. Firstly, weights w_{ijk} are initialised with a random number using MCS-MCMC algorithm.

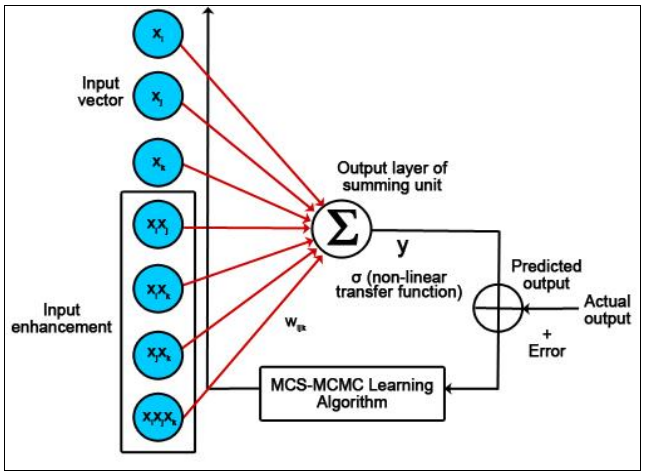


Fig.2. The Architecture of FLNN-MCMC

The standard FLNN structure, consisting of inputs, weight parameters, biases, and the activation function of the output node, is transformed into the objective function as shown in Fig.2. The standard FLNN architecture (weight and biases) is transformed into the objective function in the initial process. This objective function is then fed into the MCS-MCMC algorithm along with the training data in order to look for optimum weight parameters which can decrease the objective function while minimising the error rate. The weight changes are then tuned based on the error calculation (the difference between actual and predicted outputs) by the MCS-MCMC algorithm. The optimal weights set obtained from the training phase will then be used on unseen data for the prediction task.

3.2. *MCS-MCMC Algorithm for PSNN (PSNN-MCMC)*

In MCS, Lévy flight is used to search and exploit which signifies a possible solution to the optimisation problem. However, such Lévy flight suffers from the issue, whereas some information is utilised indirectly without analysis. This makes MCS actively lead to inefficient and failure of the algorithm. The idea of substituting the MCS with MCMC is like twofer; helping the algorithm converges rapidly to the best position while simultaneously performing full-dimensional jumps at each iteration. The algorithm is used for weight initialisation and weight update (replacing the current BP algorithm in the standard PSNN). Therefore, the subject matters of implementing MCS-MCMC algorithm into standard PSNN is to search the optimal weights values of standard PSNN. Fig.3 illustrates the architecture of standard PSNN with the added of MCS-MCMC algorithm. That MCS-MCMC algorithm substitutes the current BP algorithm which is commonly found in the standard PSNN model.

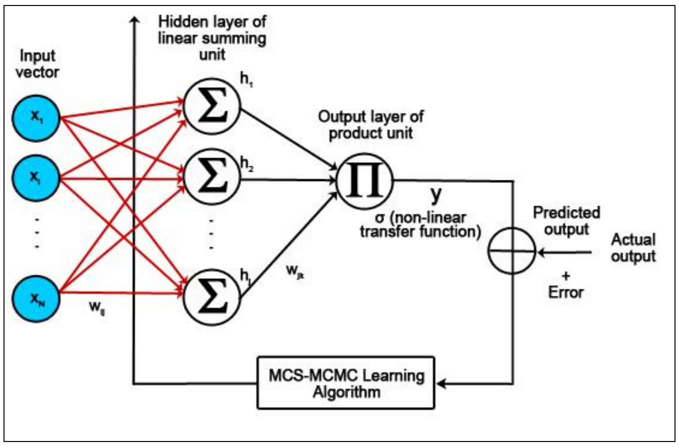

Input vector
Hidden layer of linear summing unit
x_1
x_i
x_N
w_{ij}
h_1
h_2
h_l
w_{jk}
Output layer of product unit
y
σ (non-linear transfer function)
Predicted output
Actual output
+
Error
MCS-MCMC Learning Algorithm

Fig.3. The Architecture of PSNN-MCMC

$x_1, x_2, \ldots x_n$ denotes input vectors, w_{ij} denotes adjustable weights for input vectors to linear summing unit, σ is the non-linear activation function, $h_1, h_2, \ldots h_l$ indicates the summing units, y is the output node and w_{jk} is the fixed weights from linear summing units to the output layer. At the first place, weights w_{ij} from input vector to the linear summing unit h_l is initialised with a random number using MCS-MCMC algorithm. Those random weights are evaluated from layer-to-layer to improve the searching strategies to get the optimal weights set. The weights and biases of PSNN are preserved as the optimisation parameters. These parameters and the optimisation criterion (network error criterion) are transformed into the objective function. The optimisation criterion serves to minimise the objective function's value.

SI algorithms have been an interesting alternative for providing satisfactory solutions. Almost all SI algorithms tend to fit for global optimisation. As this approach uses a particular trade-off of randomisation and local search, it enables some objective functions or error functions to be reduced or maximised. Furthermore, this approach employs high-level techniques for exploration and exploitation of the search space in which the duration of its step gradually becomes much longer. Therefore, it is indirectly making the SI scalable and flexible to handle different aspects simultaneously. All these benefits make the SI algorithm suited for weight optimisation.

4. Experimental Design and Simulation Results

The MCS-MCMC algorithm was developed using MATLAB 7.10.0 (R2010a) on Intel® Core ™ i7 CPU environment. Four (4) dataset used in these experiments.

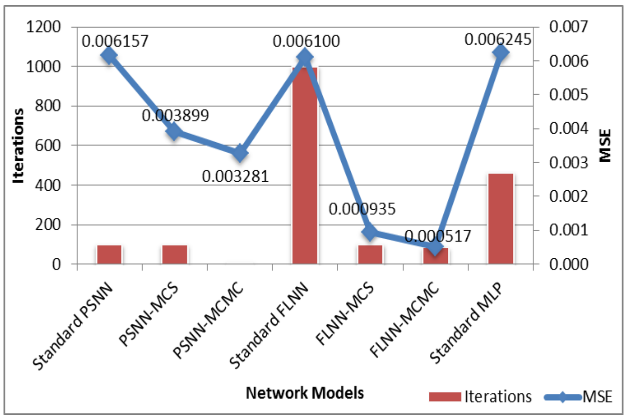
Temperature: The dataset was collected from Central Forecast Office, Malaysian Meteorological Department (MMD) [39]. Each dataset consists of 1813 that covers over year of 1992 until 2009.
Ozone: Ozone data were collected from UCSI consists of 480 instances.
Gold Close Price: The dataset contains the weekly gold close price per ounce from 01/04/1968 to 18/04/2016 (measured in US Dollar). The dataset was retrieved from the Deutsche Bundesbank Data Repository and being exported out to Kaggle Repository.
Bitcoin Closing Price: The following dataset is the daily closing price of bitcoin from the 27/04/2013 to the 03/03/2018. The source of the data is coinmarketcap.com, also available in Kaggle Repository.

The datasets were segmented into 50:25:25 subsets; for training, validation and testing. The performance metrics such as iterations and Mean Squared Error (MSE) were calculated to test the performance of all the networks. Based on the evaluation criteria, the less value indicates better performance.

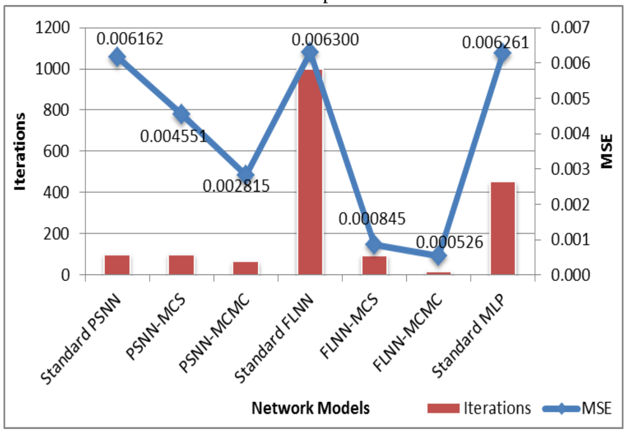
The development of the right architecture requires several steps: determining the number of layers, the amount of neurons to be used in each layer and selecting the necessary neurons' transfer function. Therefore, the parameterisation in this research work includes defining the networks combination of three (3) different numbers of input nodes ranging from 5 to 7, hidden layer/higher order terms from 2 until 5 and a single neuron for the output layer. The performance of the MCS-MCMC algorithm were evaluated on seven (7) different network architectures which are standard PSNN, PSNN-MCS, PSNN-MCMC, standard FLNN, FLNN-MCS, FLNN-MCMC and standard MLP. The maximum iterations are set at 3000 for all networks.

4.1. *Temperature Dataset*

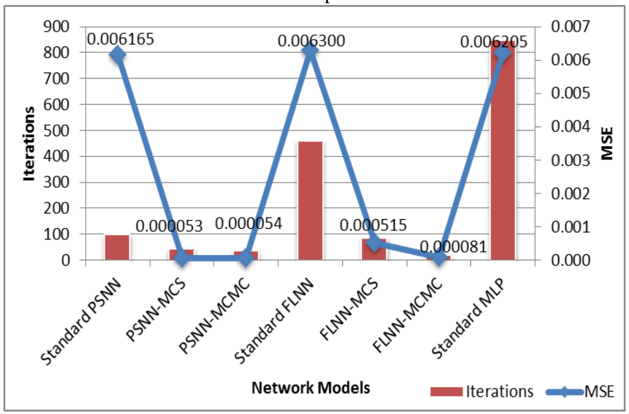
Fig.4 graphically shows the performance comparison for all the networks on Temperature dataset. According to the results plotted in Fig.4, the first, second and third ranks are FLNN-MCMC, FLNN-MCS and PSNN-MCMC for 5 and 6 inputs. While FLNN-MCMC, FLNN-MCS and PSNN-MCS for 7 inputs. From these results, it is said that the incorporation of MCS-MCMC algorithm into both FLNN and PSNN help to minimise the error rate thus assists the networks to converge quickly. As it has been pointed out, FLNN-MCMC shows the least MSEs for most of the inputs. Hence, by having the least MSE, it unites both the variance of the estimator and its bias on how far off the average estimated value went from the truth. In addition, if looking at the behaviour of Temperature data itself, it has positive skewness. This skewness shows that the data has strong influence/fluctuation that is sufficiently stable to deal with the network models integrated with MCS-MCMC algorithm.



a. 5 Inputs



b. 6 Inputs

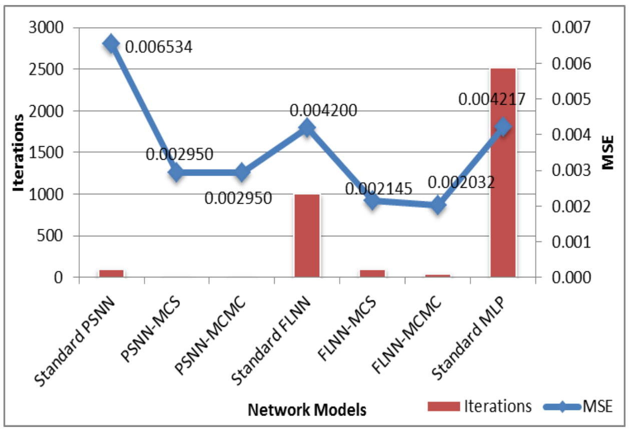


c. 7 Inputs

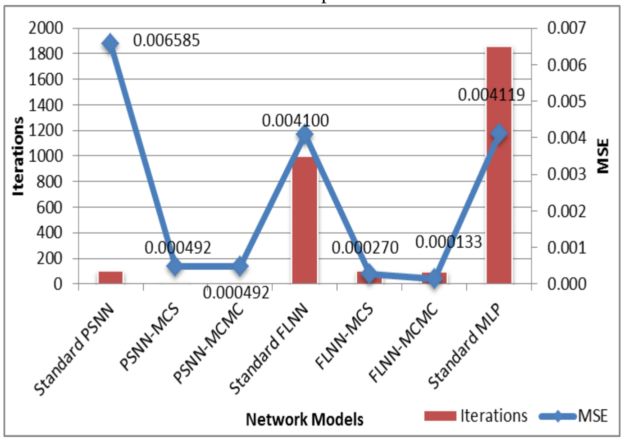
Fig.4. Performance Comparison on Temperature Dataset

4.2. Ozone Dataset

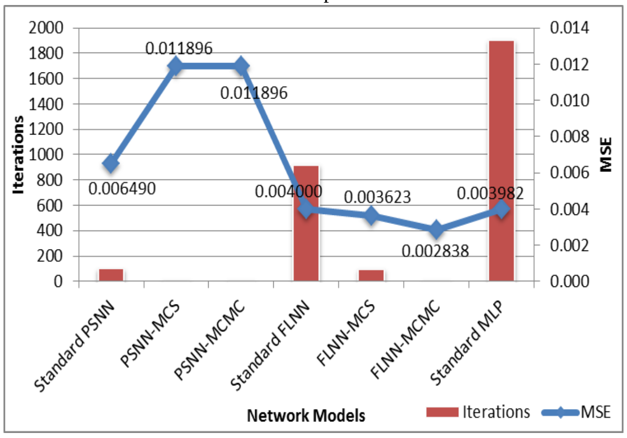
Fig.5 depicts the performance comparison for all the networks on Ozone dataset inputs ranging from 5 to 7. Indubitably, MCS-MCMC has been shown to learn the data with a comparable network size with FLNN and PSNN, with a smaller size when compared to other networks. For the inputs covering 5, 6 and 7 inputs, FLNN-MCMC outperformed the other network models with a value of 0.002032, 0.000133 and 0.002838. Throughout the results, the FLNN-MCMC shows the least MSE for the three (3) input combinations. This way, it can be iterated that the MSE are able to assess the quality of sample data to estimate the function mapping arbitrary inputs to a sample of values of certain random variables.



a. 5 Inputs



b. 6 Inputs

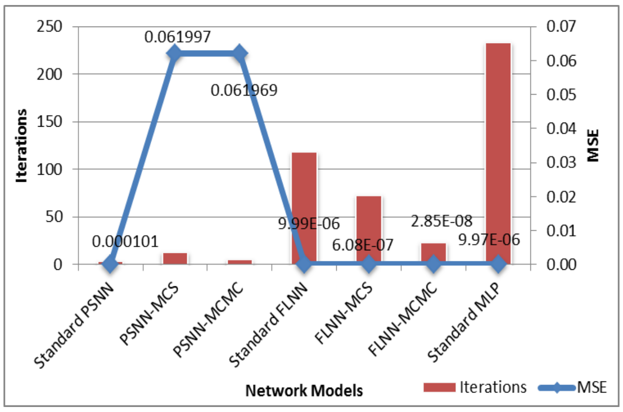


c. 7 Inputs

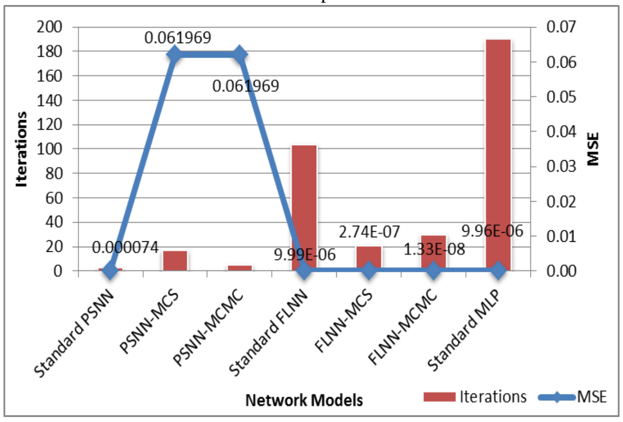
Fig.5. Performance Comparison on Ozone Dataset

4.3. Gold Close Price Dataset

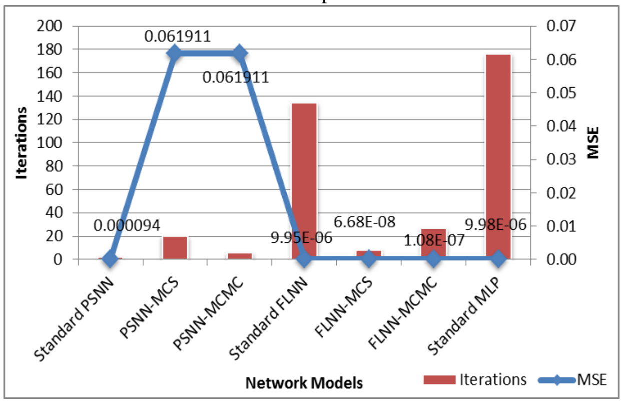
As can be seen from Fig.6, the networks with the presence of MCS-MCMC algorithm (FLNN-MCMC and PSNN-MCMC) made the least amount of MSE for ranges of inputs on Gold Close Price dataset when compared to other network models. All the networks show a very small degree of deviation from the means, as the indicator of coherent behaviour they generate. When referring to Fig.6, the FLNN-MCMC leads the rank. Upon inputted with 5 and 6 inputs, FLNN-MCS and standard MLP follow this on the second and third place. However, the rank changes as FLNN-MCS wins the first position when 7 inputs were fed up. This may happen somehow as the network combination (number of inputs, data partitions, etc.) plays a role in minimising the error rate. The lowest error value of 2.85E-08 observed in FLNN-MCMC (refer to Fig.6(a)).



a. 5 Inputs



b. 6 Inputs

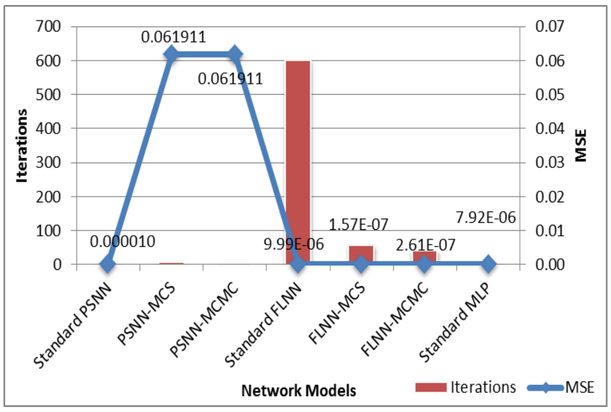


c. 7 Inputs

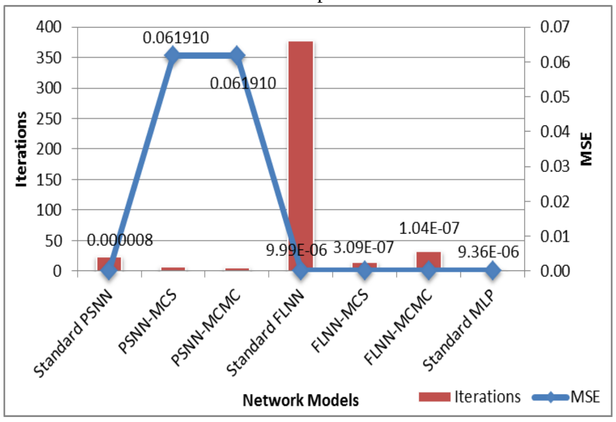
Fig.6. Performance Comparison on Gold Close Price Dataset

4.4. *Bitcoin Closing Price Dataset*

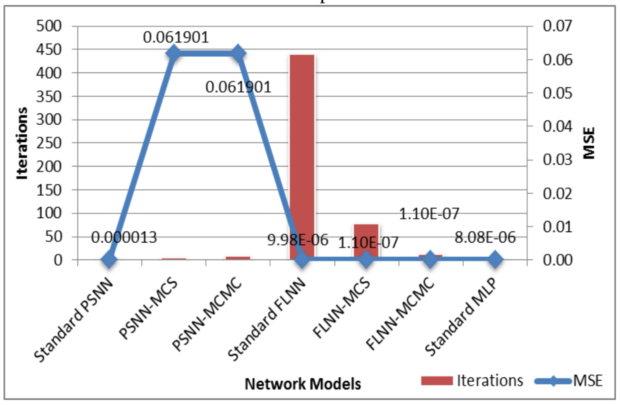
Fig.7 illustrates the iterations against MSE of each network models. The FLNN-MCMC is discovered to be more precise. It can be seen on the figures that the first, second and third ranks are dominated by FLNN-MCMC. FLNN-MCS and standard MLP on all inputs ranging from 5 to 7 (except for input 6, 50:25:25). This is due to the Bitcoin Closing Price data trend itself, sometimes increasing in prices, dropping prices, or sideways ranging. Seemingly, based on the results, the network performance in which the learning technique was substituted by MCS-MCMC algorithm is surpassing the standard MCS algorithm and the current BP algorithm.



a. 5 Inputs



b. 6 Inputs



c. 7 Inputs

Fig.7. Performance Comparison on Bitcoin Closing Price Dataset

5. Discussions

In this section, some of the issues raised by the comparison of different NN are addressed. As the results presented previously cover broad and extensive simulations, this section elaborates the observation derived from the whole of the experimental results. Therefore, this section provides the discussion on all four (4) datasets.

5.1. Model Performances based on Ranking

This tables cover four (4) datasets, inputs ranges from 5 to 7 and seven (7) network models. Table 1 indicates the overall rank for Temperature on all networks.

Table 1. Overall Rank for Temperature on all Networks

Inputs	Standard PSNN	PSNN-MCS	PSNN-MCMC	Standard FLNN	FLNN-MCS	FLNN-MCMC	Standard MLP
5	7	4	3	5	2	1	6
6	5	4	3	7	2	1	6
7	5	1	2	7	4	3	6
Mean Rank	5.67	3.00	2.67	6.33	2.67	1.67	6.00
Overall Rank	5	4	2	7	2	1	6

According to Table 1, FLNN-MCMC outperforms the other network models by having the highest average rank. This is followed by FLNN-MCS and PSNN-MCMC, both in the second rank. Basically, those swarm-based algorithm helps to overcome the drawbacks of the current BP algorithm. Table 2 shows the overall rank for Ozone dataset on all networks.

Table 2. Overall Rank for Ozone Dataset on all Networks

Inputs	Standard PSNN	PSNN-MCS	PSNN-MCMC	Standard FLNN	FLNN-MCS	FLNN-MCMC	Standard MLP
5	7	4	3	5	2	1	6
6	7	4	3	5	2	1	6
7	5	7	6	4	2	1	3
Mean Rank	6.33	5.00	4.00	4.67	2.00	1.00	5.00
Overall Rank	7	5	3	4	2	1	5

The findings from Table 2 showed that the proposed MCS-MCMC algorithm attained the highest rank when incorporated with FLNN-MCMC (for Ozone dataset). This is followed by FLNN-MCS falling into second place and PSNN-MCMC at the third place. The average rank lingered between the combinations of HONN with swarm-based algorithm. This implies that the combination of HONN and swarm-based algorithm matches well in coping with time series data. Table 3 tabulates the evaluation based on the average rank for the Gold Close Price dataset.

Table 3. Overall Rank for Gold Close Price Dataset on all Networks

Inputs	Standard PSNN	PSNN-MCS	PSNN-MCMC	Standard FLNN	FLNN-MCS	FLNN-MCMC	Standard MLP
5	5	7	6	4	2	1	3
6	5	7	6	4	2	1	3
7	5	7	6	4	1	2	3
Mean Rank	5.00	7.00	6.00	4.00	1.67	1.33	3.00
Overall Rank	5	7	6	4	2	1	3

Table 3 shows that when compared to other network models, the FLNN-MCMC reached the largest average rank. As the FLNN itself is conveniently used for approximating function which introduces faster convergence rate. Table 4 tabulates the rank for Bitcoin Closing Price dataset obtained from all network models.

Table 4. Overall Rank for Bitcoin Closing Price Dataset on all Networks

Inputs	Standard PSNN	PSNN-MCS	PSNN-MCMC	Standard FLNN	FLNN-MCS	FLNN-MCMC	Standard MLP
5	5	7	6	4	2	1	3
6	5	7	6	5	2	1	4
7	5	6	7	4	2	1	3
Mean Rank	5.00	6.67	6.33	4.33	2.00	1.00	3.33
Overall Rank	5	7	6	4	2	1	3

Overall, the findings in Table 4 are on the side of FLNN-MCMC. This is followed in the second rank by FLNN-MCS. It is therefore worth noting that the MCS-MCMC algorithm enables the FLNN to be trained and improves the accuracy.

5.2. The Effects of Network's Order/Hidden Nodes

In order to test the modelling capabilities and the stability of all network models, Fig.8 to 14 illustrate the MSEs over the network's order or hidden nodes covering all four (4) datasets. The performance of all the network models was evaluated based on the number of higher order terms ranging from 2 to 5 (for standard PSNN, PSNN-MCS, PSNN-

MCMC, standard FLNN, FLNN-MCS and FLNN-MCMC) while a different number of hidden nodes increased from 3 to 8 for MLP. The plots indicate that the MSEs of the network models rose when a network's order/hidden node was added to the network together with the network's growth.

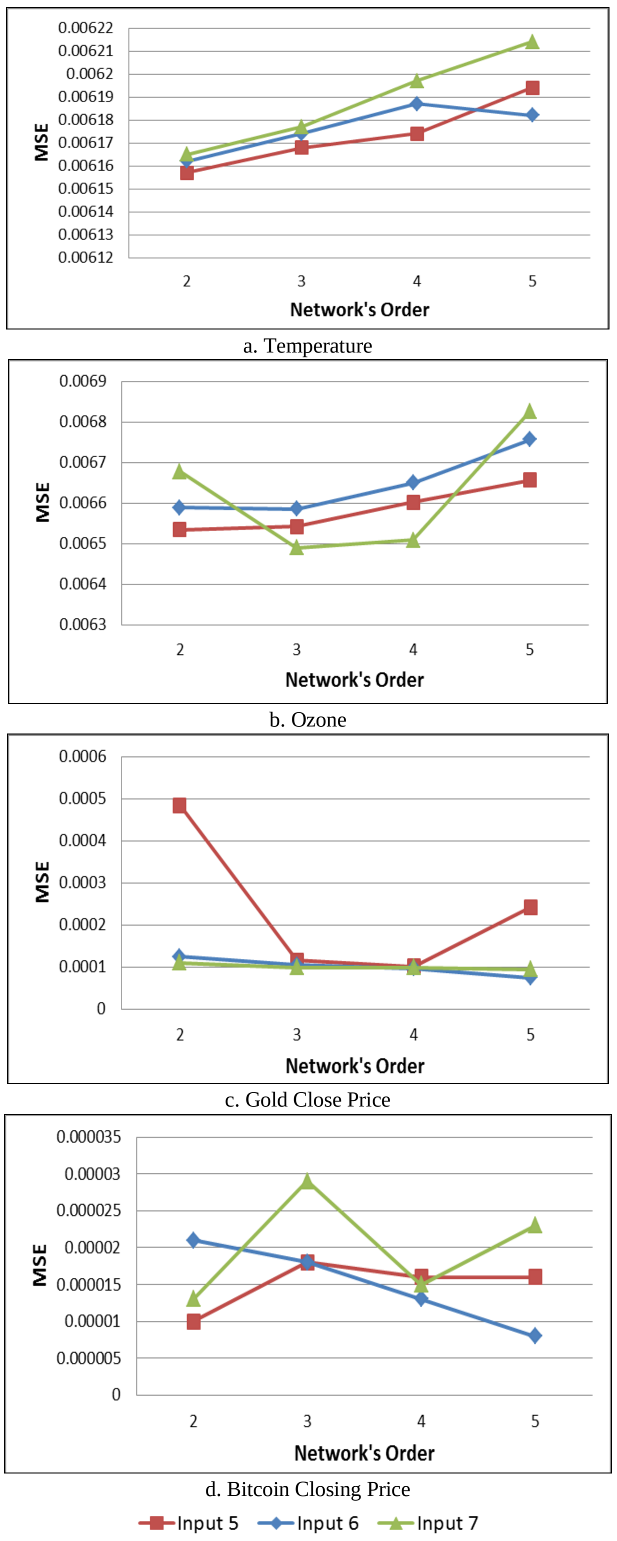
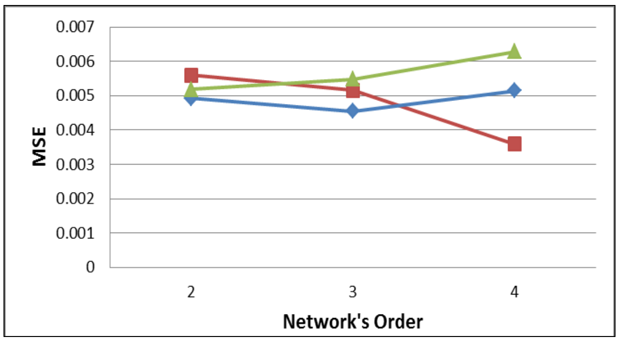


Fig.8. The Effects of Higher Order on Standard PSNN

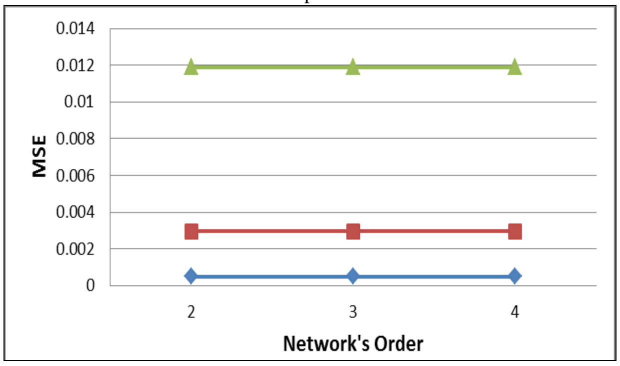
A. *Standard PSNN*

The plots in Fig.8 show that the standard PSNN learned the data steadily with the MSE continuing to decline along the growth of the network. The MSEs accelerate at constant velocity for Temperature and later on reduce. In this case, it.

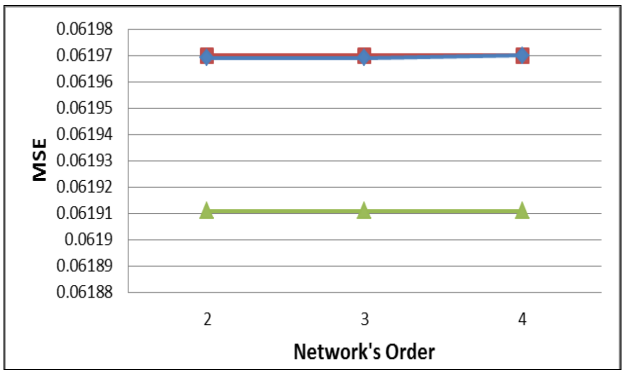
can be said that the network models reach the lowest MSE, the network models already reach almost the minimum error However, the MSE rises at some point. As the network's order increases, the network is already saturated. The same applies to Ozone as well input 7 when the network's order reached 5. For Gold Close Price and Bitcoin Closing Price show a 'zig-zag' pattern, implies that the datasets do not have a definite pattern. As far as Gold Close Price is concerned, iterations for input 7 have accomplished the least.



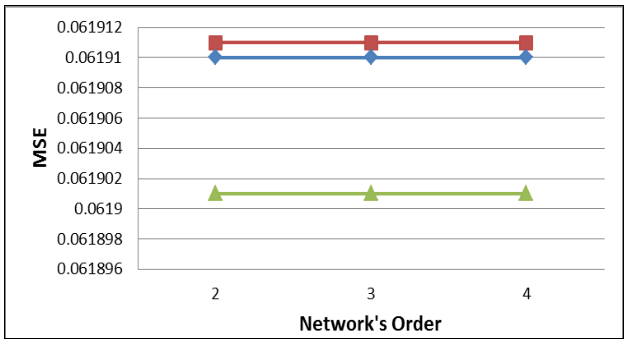
a. Temperature



b. Ozone



c. Gold Close Price

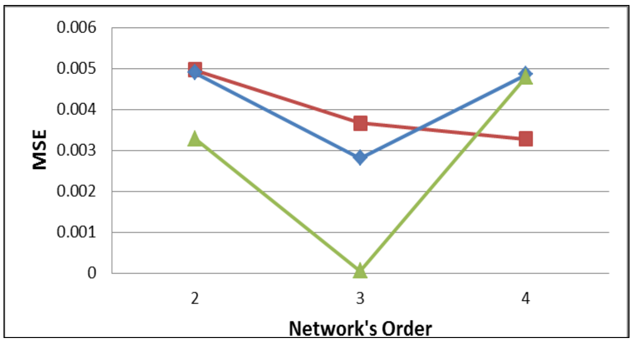


d. Bitcoin Closing Price

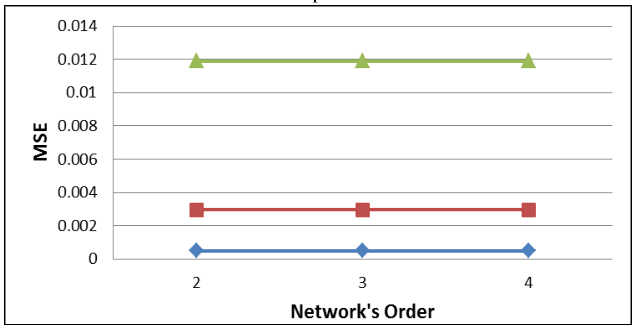
Fig.9. The Effects of Higher Order on PSNN-MCS

B. *PSNN-MCS*

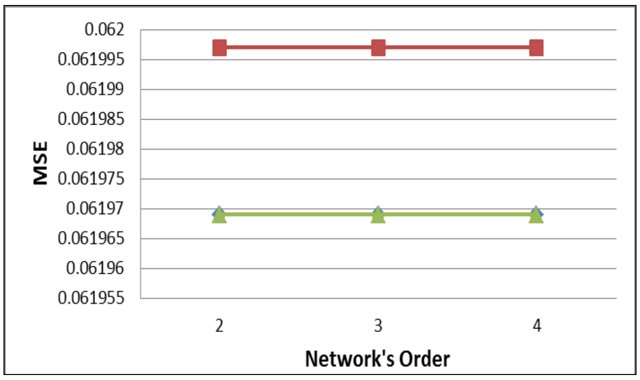
As for PSNN-MCS, higher order terms for all datasets show no constructive differences as the network's order increases except for Temperature. The MSE falls for Temperature when it reaches 3rd order and again increases for input 5. As for Temperature, the MSE shows ups and downs movement. This indicates that the data pattern is presently being well-learned by the PSNN-MCS. The iterations stopped when the minimum MSE reached nearly targeted error. The results of MSE against network's order are plotted in Fig.9. According to the results in Fig.9, it demonstrates that the 50:25:25 data partition is well-suited to most of the datasets.



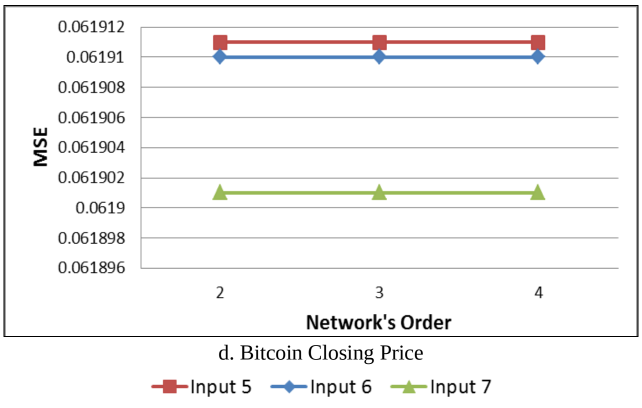
a. Temperature



b. Ozone



c. Gold Close Price

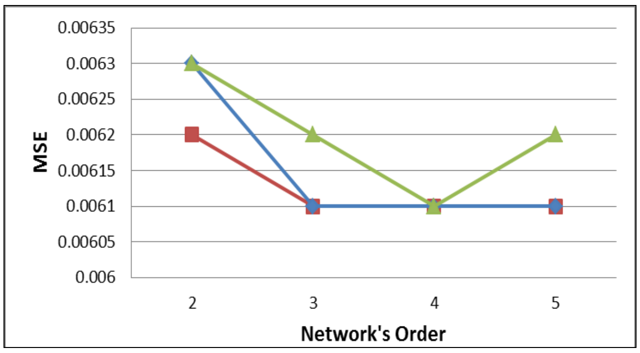


d. Bitcoin Closing Price

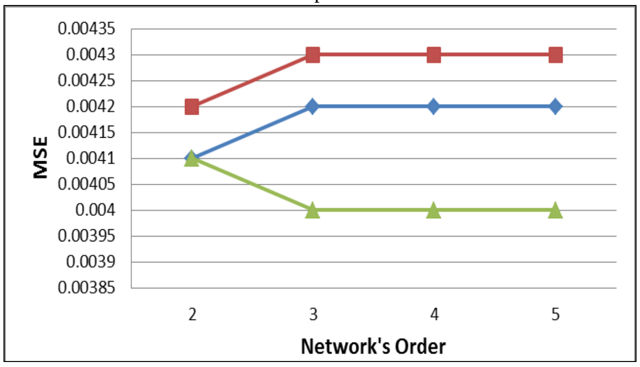
Fig.10. The Effects of Higher Order on Standard PSNN-MCMC

C. *PSNN-MCMC*

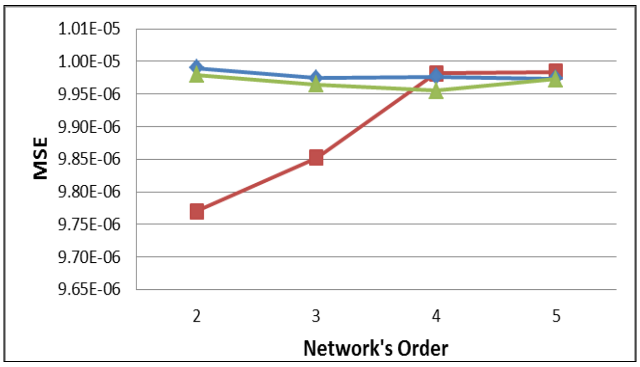
As far as PSNN-MCMC is concerned, the MSE slowly increases against the network's order for most datasets and is smoothly close to par. Therefore, the datasets can be said to be sufficiently stable and work well with the PSNN-MCMC. It is however, indicates a distinction when inputting the PSNN-MCMC with Temperature. To sum up, it is revealed, according to the results based on Fig.10, that the PSNN-MCMC provides significant improvement in all datasets where the PSNN-MCMC can improve accuracy.



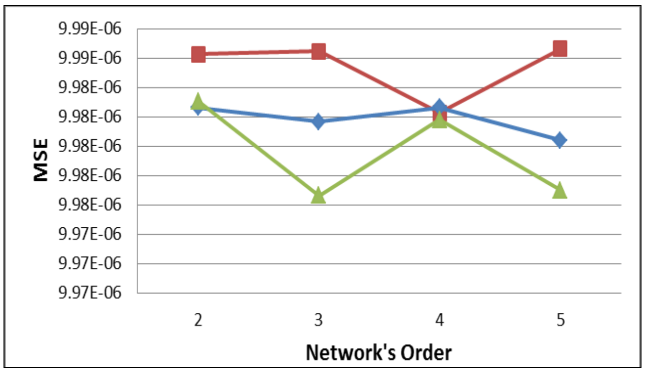
a. Temperature



b. Ozone



c. Gold Close Price



d. Bitcoin Closing Price

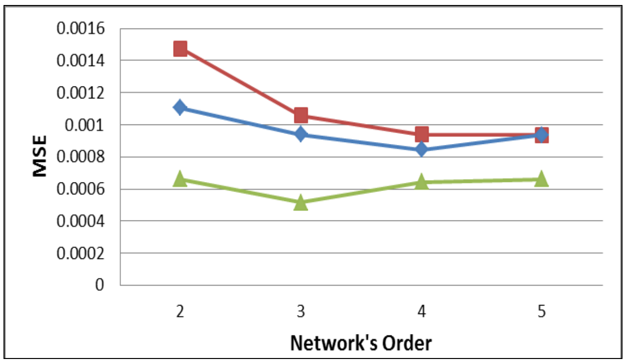
Fig.11. The Effects of Higher Order on Standard FLNN

D. *Standard FLNN*

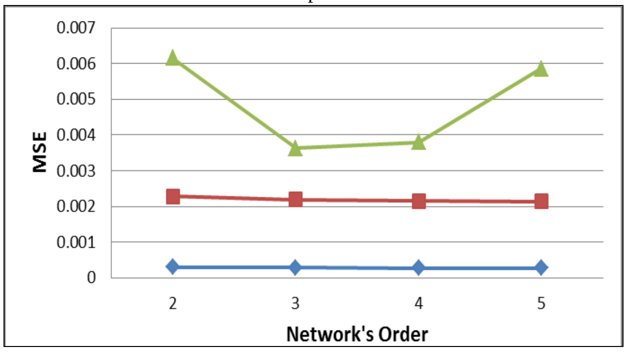
According to Fig.11, Ozone datasets demonstrate a constant and steady increment along the network's order. Inputs 5 and 6 show a sequence of minimising MSE through the network's order for the Temperature dataset. While inputs 6 and 7 show a substantial decrease in the network's order. For Gold Close Price, it shows smaller combinations of network architecture. The combinations help in achieving the lowest MSEs. The patterns are constantly accelerating in a stable state for Sunspot dataset. Input 7 provides the smallest MSE among all the data partitions for Bitcoin Closing Price. Thus, it can be concluded that a network with smaller combination of input-output gives lowest MSE, which indirectly increase the performance rate.

E. *FLNN-MCS*

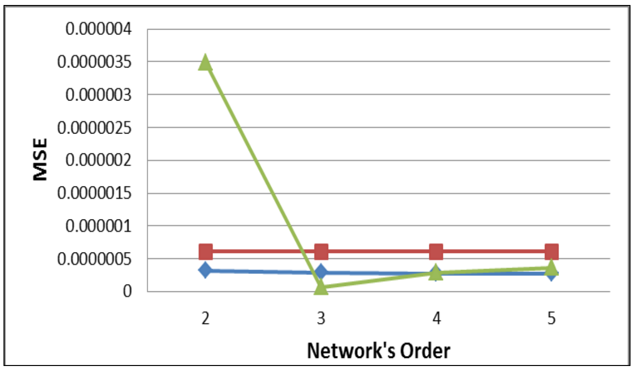
Bitcoin Closing Price and Gold Close Price datasets show a rapid deceleration and fall at a steady speed in the opposite direction when the datasets are inputted with inputs 5 to 7. For Temperature and Ozone, the line is not too steep for all inputs expect for input 7. Accordingly, it indicates that when input 7 is fed into FLNN-MCS, the MSEs decrease. This four (4) datasets show that input 5 dominates the least MSE. When comparing the 50:25:25 data partition, input 6 dominates on Ozone datasets. While for input 7 works on Temperature and Bitcoin Closing Price. These differences indicate that when dealing with nonlinear data, the FLNN-MCS sometimes was unable to learn the data pattern as the datasets were covering on time series dataset. Meaning that, the data pattern depends on various factors such as seasonality, trends, cyclicality, projects, events, and many more. Overall, the FLNN-MCS structures that deliver the best average results on the lowest MSEs were mostly carried out in 4th order on input 7 for most the four (4) datasets.



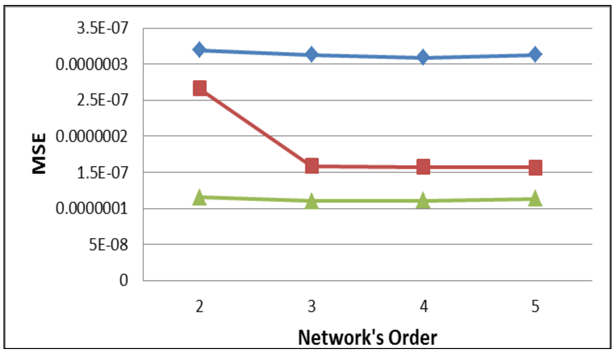
a. Temperature



b. Ozone



c. Gold Close Price

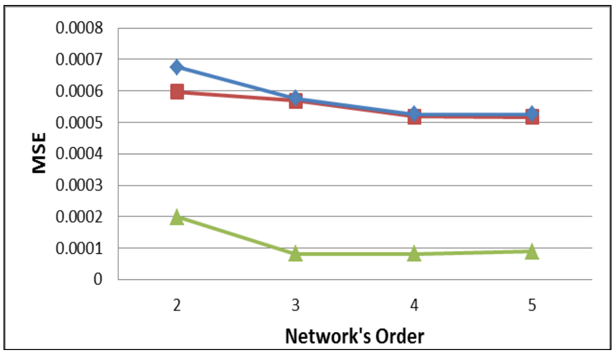


d. Bitcoin Closing Price

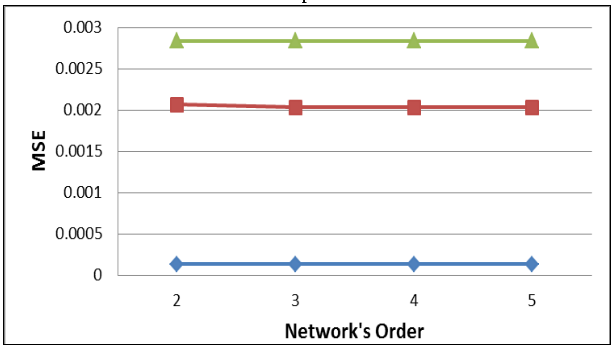
Fig.12. The Effects of Higher Order on FLNN-MCS

F. FLNN-MCMC

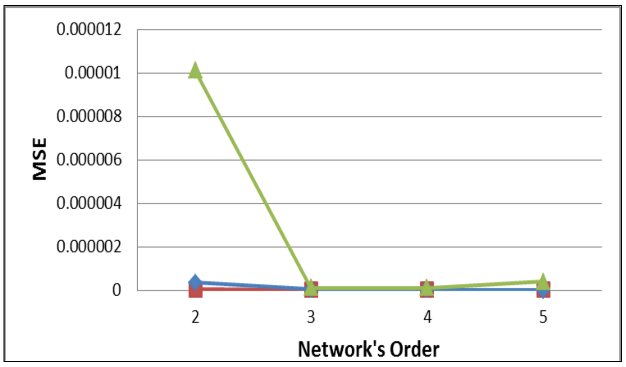
From Fig.13, the datasets show a significant drop in a number of MSEs throughout the network's order. However, when the network's order increases to 5th order, the MSE begins to increase as for Temperature and Gold Close Price datasets. This is because, the network started to become more complicated in structure when the order reached 5 or greater. As a result, the training process becomes slower. This may lead to overfit while the network begins to memorise training data, yet, it has not learned to generalise to new situations. Thus, resulting in not-so-accurate results. However, different patterns have been found in Temperature. Throughout the figures, input 7 dominates the low MSE sequence. When comparing Bitcoin Closing Price dataset, the MSE degradation pattern is the similar. The FLNN-MCMC giving the best average result were mostly realised with input 5 and 4th order terms which gives the lowest MSE for most of the datasets.



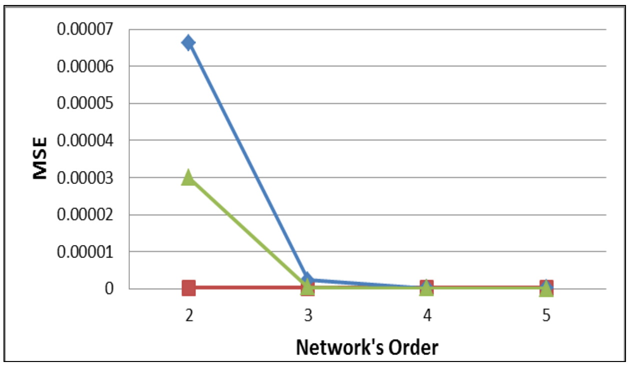
a. Temperature



b. Ozone



c. Gold Close Price

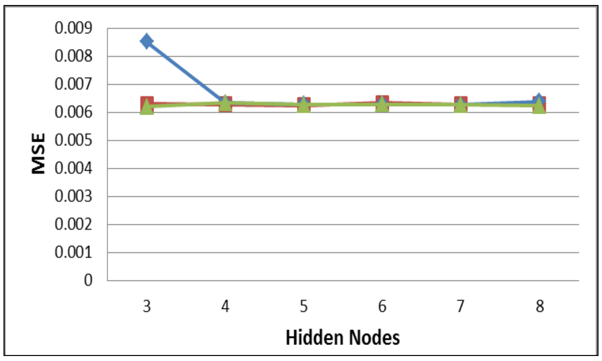


d. Bitcoin Closing Price

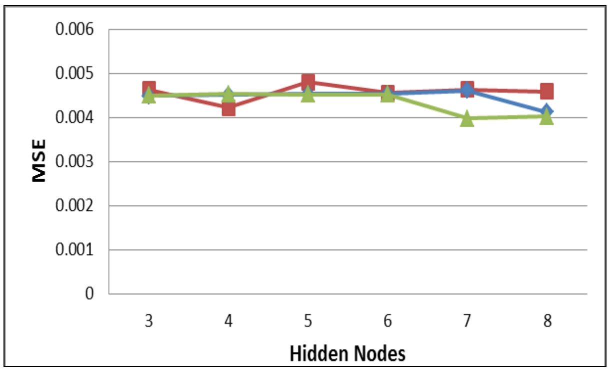
Fig.13. The Effects of Higher Order on FLNN-MCMC

G. Standard MLP

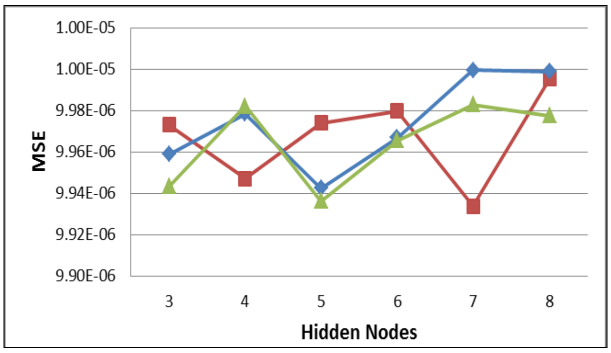
Looking at standard MLP (refer to Fig.14), the plots show a ‘zig-zag’ line that for most datasets squiggled along the x-axis (except for Temperature). This indicates that not all of the values on the number line used are included in the data displayed. In other words, the datasets do not have a clear pattern. However, the plotting on Temperature decelerates slowly and at some stage reached into a stable state. Inputs 5 and 7 demonstrate a shallow MSE result as the hidden nodes rise from 3 to 8. This shows that the behaviour of the Temperature is an untrended pattern of seasonal information. This implies that the seasonality is always fixed, and the frequency is known. Together with Temperature, the other datasets also yield results where some of the inputs yield shallow results while some offer a pattern of ‘zig-zag’. These phenomena because of the time series data behaviour itself that show ups and downs.



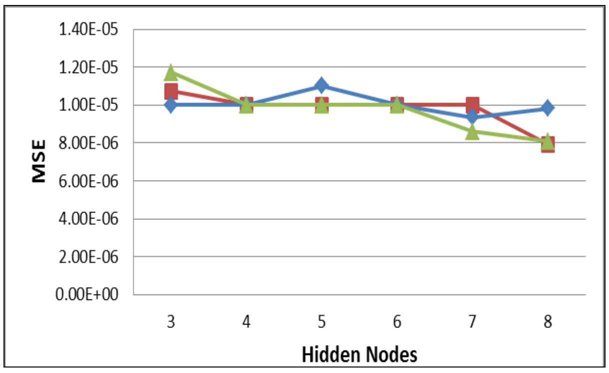
a. Temperature



b. Ozone



c. Gold Close Price



d. Bitcoin Closing Price

Fig.14. The Effects of Hidden Nodes on Standard MLP

Resuming the experiments, two (2) major areas of improving the variations of HONN are highlighted: (a) the use of MCS-MCMC with the FLNN and PSNN in weight initialisation in order to improve the network performance in terms of lower error, and topology optimisation, and (b) the replacement of current BP learning algorithm with MCS-MCMC to minimise the error. By incorporating these two (2) techniques, it preserved all the advantages of MCMC random walk while maintaining the excellent side of MCS. The significance of combining both MCS and MCMC is that the constructed learning algorithm gives higher polynomial convergence rates as the MCMC random walk itself uses local movements based on certain types of target density which leads to better algorithms qualitatively.

The convergence analysis of MCS-MCMC incorporated with HONN's network varies on a different number of inputs data, various number of higher order terms for a target function, and results were drawn. Hence, the experiments covering varying higher order/hidden nodes are needed to test the stability of the network model. Overall, the results show that network models with a swarm-based algorithm perform and shows a stable state.

6. Conclusion

The current research includes trials of MCS-MCMC algorithm on various network models. From the results, it is proved that, in this research, it is affirmative that the networks with MCS-MCMC algorithm were well generalised and showed least error compared to other network models, which could represent nonlinear function. MCS-MCMC's existence as the algorithm that replaces the current BP algorithm enabled fast and rapid training. A significant advantage of the MCS-MCMC is that with little user interference, the algorithm can automatically adjust better parameters to find excellent parameter values. This process can be accomplished through Markov chain mixing and a functional of interest integrated autocorrelation. Ultimately, it improves network performance and increase the accuracy by achieving the highest average ranking (revisit Section IV). Even though the possibilities of practical applications for MCS-MCMC have been shown, there are still some essences to be explored for the reliability of the MCS-MCMC algorithm. Therefore, recommendations regarding further development of this research work are needed. This research is focusing on using univariate dataset. Further, comparisons with other SI algorithms and techniques that can integrate knowledge in the form of constraints on multivariate data are suggested, so that data can be properly analysed.

Acknowledgment

The authors would like to thank Ministry of Education Malaysia, Universiti Tun Hussein Onn Malaysia, and Research Management Centre for funding this research under the Fundamental Research Grant, Vote No. 1651.

References

[1] Husaini, N.A., et al. *Jordan pi-sigma neural network for temperature prediction*. in *International Conference on Ubiquitous Computing and Multimedia Applications*. 2011. Springer.

[2] Holland, J., *Adaptation in natural and artificial systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. 1992, MIT Press: Ann Arbor, USA.

[3] Goldberg, D., *Genetic algorithms in search, optimization and machine learning*. 1989, Boston, USA: Addison Wesley.

[4] De Jong, K., *Analysis of the behavior of a class of genetic adaptive systems*. 1975, University of Michigan: Ann Arbor, MI.

[5] Fogel, L., A. Owens, and W. MJ, *Artificial intelligence through simulated evolution*. 1966, Chichester, UK: John Wiley.

[6] Storn, R. *Differential evolution design of an IIR-filter*. in *IEEE International Conference on Evolutionary Computation*. 1996. Nagoya.

[7] dos Santos Coelho, L. and D.L. de Andrade Bernert, *An improved harmony search algorithm for synchronization of discrete-time chaotic systems.* Chaos, Solitons & Fractals, 2009. **41**(5): p. 2526-2532.

[8] Karaboga, D., B. Akay, and C. Ozturk. *Artificial bee colony (abc) optimization algorithm for training feed-forward neural networks*. in *Proceedings of the 4th international conference on Modeling Decisions for Artificial Intelligence, ser. MDAI '07*. 2007. Springer-Verlag.

[9] Aljarah, I., H. Faris, and S. Mirjalili, *Optimizing connection weights in neural networks using the whale optimization algorithm.* Soft Computing, 2018. **22**(1): p. 1-15.

[10] Mason, K., J. Duggan, and E. Howley. *Neural network topology and weight optimization through neuro differential evolution*. in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 2017.

[11] Salimans, T. and D.P. Kingma. *Weight normalization: A simple reparameterization to accelerate training of deep neural networks*. in *Advances in neural information processing systems*. 2016.

[12] Yang, X.S. and S. Deb. *Cuckoo search via Lévy flights*. in *Proceedings of the World Congress on Nature & Biologically Inspired Computing (NaBIC '09*. 2009. India: IEEE Publications.

[13] Gandomi, A., X.-S. Yang, and A. Alavi, *Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems.* Engineering with Computers, 2012. **29**(1): p. 17-35.

[14] Sickel, J., K. Lee, and J. Heo, *Differential evolution and its applications to power plant control*, I. International Conference on Intelligent Systems Applications to Power Systems, Editor. 2007. p. 1-6.

[15] Zhang, J., Y. Zhang, and R. Gao. *Genetic algorithms for optimal design of vehicle suspensions*. in *IEEE International Conference on Engineering of Intelligent Systems*. 2006.

[16] Kaveh, A. and T. Bakhshpoori, *Optimum design of steel frames using cuckoo search algorithm with Levy flights,Structural Design of Tall and Special Buildings*. 2011.

[17] Valian, E., S. Mohanna, and S. Tavakoli, *Improved cuckoo search algorithm for feed forward neural network training.* International Journal of Artificial Intelligence & Applications, 2011. **2**(3): p. 36-43.

[18] Walton, S., et al., *Modified cuckoo search: A new gradient free optimisation algorithm.* Chaos, Solitons & Fractals, 2011. **44**(9): p. 710-718.

[19] Husaini, N.A., R. Ghazali, and I.T.R. Yanto. *Enhancing modified cuckoo search algorithm by using MCMC random walk*. in *2016 2nd International Conference on Science in Information Technology (ICSITech)*. 2016. IEEE.

[20] Saremi, S., S. Mirjalili, and A. Lewis, *Grasshopper optimisation algorithm: theory and application.* Advances in Engineering Software, 2017. **105**: p. 30-47.

[21] Sun, J., C.-H. Lai, and X.-J. Wu, *Particle swarm optimisation: classical and quantum perspectives*. 2016: Crc Press.

[22] Dems, K. and Z. Mroz, *Variational Approach by Means of Adjoint Systems to Structural Optimization and Sensitivity Analysis--II.* International Journal of Solids and Structures, 1984. **20**: p. 527-552.

[23] Chenasis, D., *Existence of a Solution in a Domain Identification Problem.* Journal of Mathematical Analysis and Applications, 1975. **52**: p. 189-219.

[24] Fujii, N. *Second Variation and Its Application in a Domain Optimization Problem*. in *Proceedings of the 4th IFAC Symposium on Control of Distributed-Parameter Systems*. 1986. Pergamon, Oxford, England.

[25] Fujii, N. *Existence of an Optimal Domain in a Domain Optimization Problem*. in *Proceedings of the 13th IFIP Conference on System Modelling and Optimization*. 1988. Springer-Verlag, Berlin, Germany,.

[26] Fujii, N., *Lower-Semicontinuity in Domain Optimization Problems.* Journal of Optimization Theory and Applications, 1988. **59**: p. 407-422.

[27] Zheng, L. *An Improved Firefly Algorithm Hybrid with Fireworks*. in *Computational Intelligence and Intelligent Systems: 10th*

International Symposium, ISICA 2018, Jiujiang, China, October 13–14, 2018, Revised Selected Papers. 2019. Springer.
[28] Gülcü, Ş., et al., *A parallel cooperative hybrid method based on ant colony optimization and 3-Opt algorithm for solving traveling salesman problem.* Soft Computing, 2018. **22**(5): p. 1669-1685.
[29] Arabasadi, Z., et al., *Computer aided decision making for heart disease detection using hybrid neural network-Genetic algorithm.* Computer Methods and Programs in Biomedicine, 2017. **141**: p. 19-26.
[30] Lahmiri, S., *A variational mode decompoisition approach for analysis and forecasting of economic and financial time series.* Expert Systems with Applications, 2016. **55**: p. 268-273.
[31] Antipov, G., M. Baccouche, and J.-L. Dugelay. *Face aging with conditional generative adversarial networks*. in *2017 IEEE international conference on image processing (ICIP)*. 2017. IEEE.
[32] Springenberg, J.T., et al. *Bayesian optimization with robust Bayesian neural networks*. in *Advances in neural information processing systems*. 2016.
[33] Wang, G.-G., et al., *A new hybrid method based on krill herd and cuckoo search for global optimisation tasks.* International Journal of Bio-Inspired Computation, 2016. **8**(5): p. 286-299.
[34] Jin, C. and S.-W. Jin, *Parameter optimization of software reliability growth model with S-shaped testing-effort function using improved swarm intelligent optimization.* Applied Soft Computing, 2016. **40**: p. 283-291.
[35] Sim, K. and W. Sun, *Ant colony optimization for routing and loadbalancing: survey and new directions.* IEEE T Syst Man Cy A, 2003. **33**(5): p. 560-572.
[36] Marinakis, Y., M. Marinaki, and G. Dounias, *Particle swarm optimization for pap- smear diagnosis.* Expert Syst Appl, 2008. **35**(4): p. 1645-1656.
[37] Mirjalili, S. and A. Lewis, *The whale optimization algorithm.* Advances in engineering software, 2016. **95**: p. 51-67.
[38] Lynn, N. and P.N. Suganthan, *Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation.* Swarm and Evolutionary Computation, 2015. **24**: p. 11-24.
[39] Malaysian Meteorological Department. *Weather Forecast*. 2010 [cited 2011 February, 18]; Available from: http://www.met.gov.my.

Authors' Profiles

Noor Aida Husaini pursued her degree at Universiti Tun Hussein Onn Malaysia (UTHM) and graduated with the Bachelor of Information Technology in 2009. Upon graduation, she was a research assistant at Software and Multimedia Centre where the research is sponsored by Ministry of Science, Technology and Innovation. She then enrolled at UTHM in 2009 for Master of Information Technology. After completing her Master, she was working as a Teaching Fellow in UTHM. In 2013, she then enrolled for the Ph.D programme in the same university.

Rozaida Ghazali is a Professor at UTHM. She received the Bachelor of Computer Science at Universiti Sains Malaysia in 1997 and continued her Masters at Universiti Teknologi Malaysia (UTM) in 2003. She received her Ph.D degree from Liverpool John Moores University in 2007. Up until now, she has published numerous book chapters, proceedings, and high impact journals. She also handled research grants and contracts of many projects' collaboration with various organisations. Her research interests include Data Classification, Swarm Intelligence, Neural Networks, Data Mining and Time Series Prediction. In 2019, she was assigned as a Director of International Office, UTHM.

Nureize Arbaiy is a senior lecturer of UTHM. She received her first degree (Computer Science) at UTM and Masters Degree (Intelligent System) at Universiti Utara Malaysia in 2001 and 2004, respectively. She has published over 20 journals and conference papers in the area Fuzzy Logic. Her representative published over 50 articles. Moreover, she was a participant of International Neural Network Society (INNS) in 2015. Her research interests include Possibilistic Decision Making, Fuzzy Random Regression Analysis, Multi Criteria Decision Making, and Time-Series Analysis. In 2019, she was assigned as a Head of Department, Postgraduate Department, Faculty of Computer Science and Information Technology (FSKTM), UTHM.

Ayodele Lasisi received his Ph.D from the FSKTM, UTHM. He currently holds a lecturing position as Lecturer I in Computer Science in the Department of Mathematical Sciences under the Faculty of Science at Augustine University, Ilara-Epe, Lagos State, Nigeria. He holds a Masters Degree with distinctions from International Islamic University Malaysia (IIUM) with specialisation in Computer Networking & Linux Administration. He completed his B.Sc. (Hons) Degree in Computer Science from Babcock University, Nigeria. His research interests include Artificial Intelligence, Computer Security, Data Communications and Networking, and Pattern Recognition. He has successfully supervised master students, authored publications in international journals and conference proceedings. He is also a member of Nigerian Computer Society (NCS), Internet Society Organisation (ISOC), and Nigerian Institute of Management (NIM).

How to cite this paper: Noor Aida Husaini, Rozaida Ghazali, Nureize Arbaiy, Ayodele Lasisi, "MCS-MCMC for Optimising Architectures and Weights of Higher Order Neural Networks", International Journal of Intelligent Systems and Applications(IJISA), Vol.12, No.5, pp.52-72, 2020. DOI: 10.5815/ijisa.2020.05.05