Time Series Prediction using Backpropagation Network Optimized by Hybrid K-means-Greedy Algorithm

J. Y. B. Tan, D. B. L. Bong and A.R.H. Rigit

***Abstract*— A multilayer perceptron with backpropagation algorithm (BP) network that has the optimal number of neurons in its hidden layer would be able to predict accurately unknown values of a time series that it is trained with. A model known as K-means-Greedy Algorithm (KGA) model which combines greedy algorithm with k-means++ clustering is proposed in this paper to find the optimal number of neurons inside the hidden layer of the BP network. Experiments performed show that the proposed KGA model is effective in finding the optimal number of neurons for the hidden layer of a BP network that is used to perform prediction of unknown values of the Mackey-Glass time series.**

***Index Terms*— Artificial Neural Network, Backpropagation Network, Greedy Algorithm, K-means++ clustering, Optimization**

I. Introduction

The BP network [1] has been implemented in many real-world applications in recent years, such as prediction of precipitation and water levels of a river [2], control of a vehicle suspension system [3], and also in image processing [4].

The most popular use of BP networks is in time series prediction, where unknown values in a time series are estimated based on information about known values of the time series [5]. According to Kumar [6], "this is because BP is easy to implement and fast and efficient to operate." A major problem in using a BP network for time series prediction however is the difficulty in obtaining the BP network with a suitable number of neurons in each existing hidden layer which is capable of producing the most accurate predictions for the time series at hand. This is because in order to obtain a BP network that generalizes well from the training data and thus produce a reasonably accurate prediction results when it is provided with time series data that it has never seen before [7], the number of neurons in the hidden layer of the BP network cannot be either too big or too small for the problem at hand [6]. If the number of neurons in the hidden layer of the BP network is too large, the BP network will only memorize the time series dataset that is given to it. As a result, it will produce excellent results if it is given data that is either similar to or exactly the same as training data, but will produce inaccurate results if it is given data that is different from the data that is used to train it [7]. In addition to that, this network will require a lot of memory and take a long time to be trained [8].

Reducing the size of the hidden layer inside a BP network would force the BP to develop general-purpose feature detectors to process the information that is input to it. This is useful for processing new inputs to the network that is never seen by the network, and thus the network performs better. [9] Zhang et al. [10] caution however that "networks with too few hidden nodes may not have enough power to model and learn the data." Again this results in a BP network that has not learned enough about the time series data, and thus produces less accurate predictions of unknown values in the time series it is trained with.

Although it is important to adjust the number of neurons in the hidden layer of the BP network so that it is just big enough to predict accurately unknown values of a time series at hand, unfortunately the task of finding the suitable number of neurons in the hidden layer of the BP network is not easy. This is because local minima, architectures that would enable the BP network to produce small errors but not the minimum error, do exist as there is a significant variation in the accuracy of the prediction depending on the number of hidden neurons in the BP network. Due to the presence of these local minima, it is easy for the search of the optimum number of neurons in the hidden layer of the BP network to become trapped in the local minima. [11] Therefore there is no guarantee that the global optimum (the architecture that will produce the minimum error between the actual data and the output of the network) can be found.

There are no rules that provide a definitive size of the hidden layer in a BP network for a particular type of problem [12]. In view of this problem, several researchers proposed methods to quickly find the optimal architecture that will produce highly accurate prediction of the time series at hand. Some of these methods start with having a large number of neurons in the hidden layer of the BP network and then utilize some heuristic to identify and eliminate parameters that are considered unnecessary and do not contribute to better predictions. Reference [7] presents a

Manuscript received February 18, 2012; revised June 28, 2012.

J. Y. B. Tan is with Dept of Electronic Engineering, Faculty of Engineering, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia, (phone: +6019-8877930; fax: +6082-342322; e-mail: yiawbeng@yahoo.com).

D. B. L. Bong is with Dept of Electronic Engineering, Faculty of Engineering, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia (e-mail: bbldavid@feng.unimas.my).

A. R. H. Rigit is with Dept of Mechanical and Manufacturing Engineering, Faculty of Engineering, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia (e-mail: arigit@feng.unimas.my).

survey of such algorithms. However, pruning-based algorithms are time consuming and yet do not guarantee that the most optimal size of the hidden layer of the BP network at hand is obtained [8]. In addition to that, there is also the question of how large is large, i.e. the number of neurons in the hidden layer of the BP network that is considered large for the problem at hand to begin with. [13]

In addition to pruning-based algorithms, established optimization algorithms such as tabu search, [14] particle swarm optimization (PSO), [15] bee algorithm [16] and genetic algorithm (GA) [11, 17] have been implemented to find the optimal number of neurons in the hidden layer of a BP network. In general, these algorithms perform two steps in order to search for the optimal number of neurons in the hidden layer of the BP network. Firstly, a range of possible configurations of the BP network that could be useful in producing accurate results in predicting a time series is produced. Then each possible configuration is evaluated to determine whether such configuration contributes to smaller error between the output of the BP network and the actual unknown values of the time series. Configurations that result in larger errors are eliminated, ultimately leaving only the optimal configuration, which in this case is the optimal number of neurons in the hidden layer of the BP network.

An example of an optimization algorithm that has been implemented to produce an optimal number of neurons in the hidden layer of the BP network is the genetic algorithm (GA). GA is used to evaluate the fitness function of the parameters of the hidden layer in the BP network. Solutions having the best fitness functions will then be used to "breed" a new generation of solutions. Typically the "breeding" involves having two solutions exchange their "genetic material" with each other. This procedure is known as a cross-over. In addition to that, a solution can be mutated, a process in which its genetic configuration is randomly changed to create a new solution. GA has been praised for its extensive range in searching for global solutions. However, in addition to the difficulty in proving the optimality of the results of such optimizations, GA has shown to be inefficient [18]. Lin et al. [19] pointed out that this is due to the fact that "it must execute a mass of prolix and redundant iteration, and the feedback information of system cannot be taken full advantage of, which decreases solution efficiency."

Greedy algorithm [20] has been implemented in several applications such as in handwriting recognition [21] and image contour detection [22]. Greedy algorithm evaluates each possible solution to search for the best solution. It rejects solutions worse than the best known solution, while accepting and then updating the best known solution with solutions that are found to be better than the currently known best solution. Greedy algorithm is implemented in this paper to optimize the number of neurons in the hidden layer of a BP network due to its simplicity, speed and effectiveness in finding a global solution. [23] The solutions to be evaluated by the greedy algorithm are thus the number of neurons in the hidden layer of the BP network. However, using greedy algorithm does not guarantee the finding of a truly global solution [20] since it does not always produce a globally optimal solution for the problem at hand [23]. This is because since the greedy algorithm usually takes solutions that look best at the moment [24] thus it does not consider all possible solutions to the problem at hand.

The inability of the greedy algorithm to guarantee finding of the globally optimal solution is addressed in this paper by combining greedy algorithm with k-means++ clustering algorithm [25] into the proposed K-means-Greedy Algorithm (KGA) model to find the optimal number of neurons inside the hidden layer of a BP network. The proposed KGA model utilizes k-means++ clustering algorithm to limit the search space for the greedy algorithm by zeroing in on the values of the number of neurons in the hidden layer of the BP network that are more likely to lead to global minimum while discarding the values that are less likely to do so. This method will result in fewer values for the greedy algorithm to consider, thereby increasing the probability of the greedy algorithm to obtain a globally optimum solution. Once the search space for the greedy algorithm is reduced to a certain degree, the greedy algorithm will evaluate the remaining values in order to find the optimal number of neurons in the hidden layer of the BP network.

The contribution of this paper is that a model to automate the process of optimizing the BP network in prediction of time series trends has been developed. To the best of our knowledge, the proposed hybrid KGA model developed in this research is an original method that has never been used elsewhere before. We believe that the KGA model will be useful in finding the optimal number of neurons in the hidden layer for any application of prediction of unknown values in a time series quickly.

This paper is divided into several sections. The problem of finding the optimal number of neurons in the hidden layer of a BP network has been explained in this section. Section II explains the implementation of the proposed KGA model, while section III will present the methods used to verify the effectiveness of the proposed KGA model in optimizing the BP network which is trained to predict unknown values of the Mackey-Glass time series. The results and discussions of experiments to evaluate the performance of the proposed KGA model in optimizing the BP network that is trained using the Mackey-Glass time series are presented in section IV. The conclusion of the research will be included in section V.

II. Algorithm of the hybrid K-Means-Greedy algorithm (KGA) model

The steps involved in implementing the algorithm of the proposed KGA model shown in Fig. 1 are as follows:

A. Initial subdivision of the range of the values of the number of neurons in the hidden layer

Since there are local minima in finding the optimal number of neurons in the hidden layer, the proposed KGA model must therefore perform a thorough search in order to increase the chances for it to find the optimal range of values of the number of neurons in the hidden layer that is most likely to produce the most accurate prediction for the

```
WHILE range of values from the database is not small enough
    Subdivide the currently evaluated range of values into smaller ranges
    FOR t times
    • Determine number of neurons in the hidden layer of the BP network
    • Create, initialize, train and simulate the BP network with the determined number of neurons in the hidden layer
    • Record error caused by the BP network
    END
    • Create database of values and the errors caused
    • Cluster analysis on the data in the database
    • Identify cluster containing values with small errors
    • Limit the currently evaluated range of values to the identified cluster
END
Evaluate the remaining values to obtain the global minimum using greedy algorithm
Identify the value of the number of neurons in the hidden layer that gives the most accurate prediction of a time series
```

Fig. 1. Pseudocode of the algorithm of the proposed KGA model

time series. To achieve this, the values of the number of neurons in the hidden layer of the BP network N that are currently being evaluated are divided into n subdivisions. The range R of values of N inside a subdivision is obtained using (1).

$$R = \frac{N}{n} \tag{1}$$

Fifteen subdivisions are set for the initial values of N which ranges from 1 to 150 neurons. This initial range of 1 to 150 neurons is chosen in anticipation of the possibility for the need to have such a large number of neurons in the hidden layer, as some researchers [26, 27] discover that there must be at least 40 neurons in the hidden layer in order to produce the most desirable results in certain applications of BP networks.

B. Initial evaluation of the error versus number of hidden neurons by the proposed KGA model

In the previous step, n subdivisions have been established within the initial range of N which ranges from 1 to 150. The proposed KGA model will now make guesses of the range of values of N that will enable the BP network to obtain the most accurate prediction of the unknown values of the time series that the BP network is trained with. To achieve this, a subdivision x is chosen randomly from the established subdivisions for the first run. Then a value of N, denoted as N_x, is randomly taken from within the range R_x of the subdivision x. Once this is done, the BP network with N_x neurons in its hidden layer is initialized and then trained and simulated with the time series data at hand. The error E_x when the BP network has N_x neurons in its hidden layer is recorded.

C. Create database of errors versus number of hidden neurons

The steps outlined in the previous subsection are now repeated (t-1) times, with the value of t arbitrarily set to be ⅓ of the total range of N currently being evaluated by the KGA model. During each run, the number of neurons in the hidden layer of the BP network is set to a value randomly taken from another randomly chosen subdivision which is not the subdivision chosen in the previous runs. This is done so that the proposed KGA model evaluates at least a value of the number of neurons in the hidden layer from each subdivision and that particular value is chosen only once. The subdivisions chosen in the previous runs will only be chosen after one value of N is chosen from each existing subdivision.

1. Find shortest distance $D(x_i)^2$ from an observation x_i and the centroid c_i that is closest to the observation x_i
2. Choose a real number w uniformly at random between 0 and $\sum_{x \in X} D(x)^2$.
3. Find a unique integer i so that $\sum_{n=1}^{i} D(x_n)^2 \geq w > \sum_{n=1}^{i-1} D(x_n)^2$
4. Choose this observation x_i as the next cluster centroid c_i.
5. Repeat steps 1 to 4 until k cluster centroids have been formed.

Fig. 2. The pseudocode of the selection of initial values of centroids after the first centroid is chosen uniformly at random among the observations x_i

After this evaluation has been run for t times, the values of N and the corresponding errors E between the predicted values and the actual values of the out-of-sample data obtained when different values of N are used are then listed in a database.

D. Perform cluster analysis on the database

A database of errors caused by the particular number of hidden neurons has been created in the previous step. After these observations, which are the values of N and the corresponding values of E, are mapped out on a 2-dimensional plane, the database is arbitrarily partitioned into 3 clusters represented by 3 centroids using k-means++ clustering. The initial coordinates of the first centroid is chosen uniformly at random from the values of the observations. The coordinates of the remaining centroids are then determined using the process shown in the pseudocode in Fig. 2. After that, the Euclidean distance D_{ij} between each observation x_i having the coordinates (N_i,E_i) and each centroid c_j with the coordinates (x_j,y_j) is computed using (2).

$$D_{ij} = \sqrt{(x_j - N_i)^2 + (y_j - E_i)^2} \tag{2}$$

A cluster C_j represented by a centroid c_j with the coordinates (x_j,y_j) will then be populated only with observations that are closest to it. This is shown in (3).

$$C_j = \left\{ x_i : D_{ij} < D_{ij^*}, i = 1,2,...,k, j^* \neq j \right\} \tag{3}$$

The existing centroids are no longer valid to represent the clusters once all observations are successfully clustered. The coordinates of the new cluster centroids are calculated by

finding the mean of all the observations found in a particular cluster. As an example, the new centroid c'_j with the coordinates (x'_j, y'_j) is calculated using (4) to find the mean of all the observations that were assigned to cluster C_j, where the value n represents the number of x_i points that are in the cluster C_j.

$$c'_j = \frac{1}{n} \sum_{x_i \in C_j} x_i \tag{4}$$

When the coordinates of the new centroids have been determined, the process of clustering observations to the new centroids and creating new centroids from the observations in each cluster is repeated until the coordinates of the centroids no longer change. When this happens the cluster C_j^* that contains the smallest coordinate values of E is identified. This is done by identifying the cluster centroid with the smallest value of E.

E. Further subdivisions and execution of the k-means++ algorithm

The process of guessing the values of the number of neurons in the hidden layer of the BP network is now repeated, but this time limited to only the range of values within the cluster C_j^*. Each time this process is performed, the range of values of N within the cluster C_j^* is further divided into smaller subdivisions. The number of subdivisions is set so that many equally sized subdivisions would be created within the cluster C_j^*. This would mean that each subdivision would have a smaller range of the number of neurons in the hidden layer, which would improve the chances of the KGA model to find the optimal number of neurons in the hidden layer. The number of neurons in the hidden layer of the BP network is then set to the values of the guesses made by the proposed KGA model, before it is trained and simulated using the time series data at hand. K-means++ clustering algorithm is then used to cluster the resulting database of guesses and the corresponding errors in order to identify the new cluster C_j^* having the centroid with the smallest value of E.

When the range of the number of neurons in the hidden layer N is not more than 10% of the original range of N, the greedy algorithm takes over the searching process and the above mentioned process is stopped.

F. Optimize the hidden layer of the BP network

Having taken over from the k-means++ clustering algorithm, the greedy algorithm now evaluates which of the remaining values of the number of neurons N in the hidden layer is the most optimal value for the BP network for the time series that it is attempting to predict. This range of candidate values is denoted as $N_{candidate}$, where $N_{candidate} = \{N_i,\ i = 1,2,...,n\}$. First of all, the proposed KGA model designates a value from $N_{candidate}$ as the optimal value N_o. The error that is achieved when N_o is utilized is denoted as E_o. The greedy algorithm then evaluates each value of N_i. Each time this evaluation is performed, it compares the value of error E_i that is achieved using a particular value of N_i that is being currently evaluated against the error E_o which is obtained using the currently best known value of N_o. While it will replace the N_o with N_i if E_i is smaller than E_o, it will discard the value of N_i it is currently evaluating if E_i is larger than E_o. This is shown in (5).

$$N_o = \begin{cases} N_i & E_i < E_o \\ N_o & \text{otherwise} \end{cases} \tag{5}$$

Once the greedy algorithm has evaluated all possible values of $N_{candidate}$ it terminates the search and displays the number of neurons in the hidden layer, N_o that will cause the BP network to produce the smallest error E_o when it is used to perform prediction of the time series at hand.

III. Evaluating the Performance of the Proposed Hybrid KGA Model

A. Time series data used

The Mackey-Glass time series [28], a nonlinear time delay differential equation shown in (6), is utilized to evaluate the effectiveness of the proposed KGA model in optimizing the number of neurons in the hidden layer of a BP network. In this paper, the values of β, n and γ shown in (6) are set to 0.2, 10 and 0.1 respectively, and that $x(0)$=1.2. This is done so that the Mackey-Glass time series data behaves in a chaotic manner, which is a standard practice when the prediction capability of a machine-learning methodology is evaluated [29, 30].

$$\frac{dx(t)}{dt} = \frac{\beta x(t-\tau)}{1 + x^n(t-\tau)} - \gamma x(t) \tag{6}$$

B. Evaluating the prediction made by the BP network

Following the methods utilized by [29] and [30], the goal for the BP network in this paper is set to predict the value of the Mackey-Glass time series $x(t+6)$ using 4 historical values as its inputs, namely $x(t-18)$, $x(t-12)$, $x(t-6)$ and $x(t)$. The output of this BP network is $x(t+6)$. The first 500 out of the 1000 data in the Mackey-Glass time series generated with the values of t ranging from 118 to 1117 are used as training dataset, while the remaining 500 are used as out-of-sample data that is used to evaluate the generalization capabilities of the trained BP network.

C. Criteria in evaluating the proposed KGA model

The BP network must be able to accurately predict values that are not included as part of the data that is used to train the BP network [31]. Thus the proposed KGA model is evaluated on the ability of the optimized BP network to produce an accurate prediction of $x(t+6)$ that are not part of the training data once the training of the BP network is complete. Root mean squared error (RMSE) shown in (7) between the output of the BP network x_i' and the actual out-of-sample portion of the dataset x_i is the error metric used in this evaluation to determine the optimal number of neurons in the hidden layer of the BP network.

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - x_i')^2}, i = 1,2,...,N \tag{7}$$

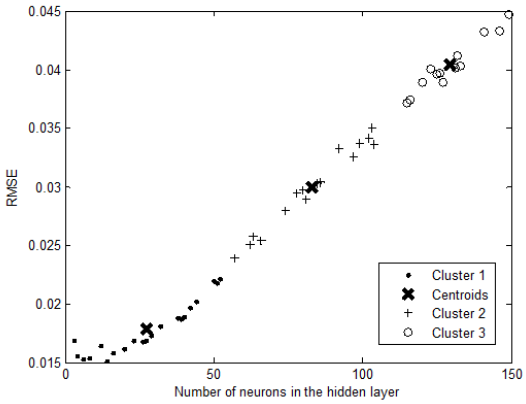

Fig. 3. Clustering of the errors and the corresponding number of neurons in the hidden layer

The weights and biases of the BP network are initialized to random values when the BP network is trained and then run with any time series data, in this case the Mackey-Glass time series data. Thus a BP network with a given value of neurons in its hidden layer will produce different values of error every time it is trained and run with the same data. Thus in this paper, after the proposed KGA model has guessed a value of the number of neurons in the hidden layer N of the BP network, the BP network is then created, initialized with random values of weights and biases, trained and then simulated for 30 times with N neurons in its hidden layer. The RMSE obtained from these 30 runs are then averaged to give a single value that represents the RMSE obtained when the BP network has N neurons in its hidden layer.

D. Comparing the findings by proposed KGA model with the findings by using an exhaustive search

In order to examine whether the proposed KGA model has indeed found the optimal number of neurons in the hidden layer, the optimal number of neurons in the hidden layer of the BP network found using the proposed KGA model is compared to the optimal number of neurons in the hidden layer of an unoptimized BP network which is found using an exhaustive search. To perform an exhaustive search for the optimal number of neurons in the hidden layer of the BP network, a BP network which is not optimized by the proposed KGA algorithm is initialized, trained and simulated using the same time series data and procedures that are used to create, initialize, train and run the BP network that is to be optimized using the proposed KGA model. This means that as the number of neurons in the hidden layer N tested by the proposed KGA model varies from 1 to 150 neurons, the unoptimized BP network is also initialized, trained and simulated with the number of neurons in the hidden layer X changed between 1 and 150 each time the BP network is executed. In addition to that, for each value of X evaluated the BP network will be run 30 times, with its weights and biases initialized to random values each time it is run. The RMSE errors obtained from these 30 runs are then averaged to give a single value of RMSE error E to represent the RMSE error obtained when the BP network has X neurons in its hidden layer.

The RMSE error E obtained when there are X neurons in the hidden layer is recorded. These results are then compared to the results of running the BP network optimized by the proposed KGA model. As an example, if the BP network optimized by the proposed KGA model determines that the optimal number of neurons in the hidden layer of the BP network is N_1 and the optimal number of neurons in the hidden layer of an unoptimized BP network found using the exhaustive search is also N_1 this means the proposed KGA model has indeed found the optimal number of neurons in the hidden layer of the BP network for prediction of unknown values in the Mackey-Glass time series.

E. Comparing the computational cost of running proposed KGA model against the computational cost using exhaustive search

The time t_{KGA} taken to implement the proposed KGA model on a system equipped with AMD Athlon64 X2 3800+ and running Windows Vista SP2 will be measured and then compared with the time t_{search} taken by the same system to perform the exhaustive search. These values of time t_{KGA} and t_{search} reflect the computational costs of implementing the proposed KGA model and the computational costs of performing an exhaustive search respectively.

For the proposed KGA model, the time t_{KGA} taken is the time taken to make a guess and then run the BP network 30 times for each guess made in order to find the average error corresponding to each guess. For exhaustive search, the time t_{search} reported is the total time taken to make 150 guesses and to run the BP network 30 times for each guess.

F. Comparing the performance of optimized BP networks against the performance of other methods in literature

By comparing the performance of the BP network optimized using the proposed KGA model on the Mackey-Glass time series against the performance of other methods found in reference [29], the effectiveness of the proposed KGA model in optimizing the BP network can be compared to other time series prediction methods.

IV. Results and Discussions

A. Results of optimization of the BP network trained using Mackey-Glass time series by the proposed KGA model

Initial guesses for the number of neurons in the hidden layer are changed from 1 to 150 neurons each time the BP network is trained and simulated with the data taken from the Mackey-Glass time series. Fifty readings, which total ⅓ of the range of the number of neurons in the hidden layer, are taken with each reading consisting of the particular number of neurons in the hidden layer of the BP network and its corresponding RMSE error. The results of plotting and the partitioning of these readings into 3 clusters are shown in Fig. 3.

The cluster marked as cluster 1 in Fig. 3 is selected by the proposed KGA model since its centroid has the smallest value of the RMSE error. Fig. 3 also shows that the range of values covered by this cluster is approximately 50, which is

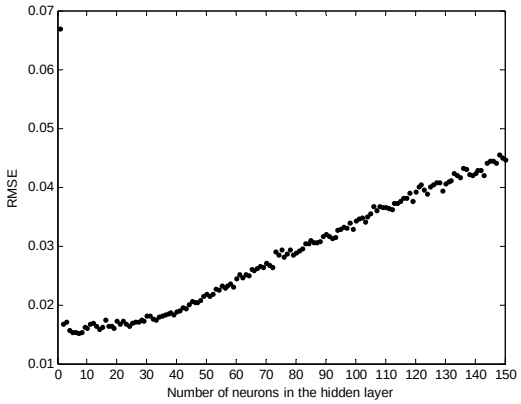

Fig. 4. Plot of the number of neurons in the hidden layer of the BP network compared to RMSE error

TABLE I
COORDINATES OF THE CLUSTER CENTROIDS AFTER CLUSTERING OF THE VALUES OF CLUSTER 1 IN FIG. 3

Cluster Represented	x-coordinates (Number of neurons in the hidden layer)	y-coordinates (RMSE)
2	12	0.015905
3	29	0.017211
1	47	0.020795

TABLE II
GUESSED VALUES OF THE NUMBER OF NEURONS IN THE HIDDEN LAYER OF THE BP NETWORK IN CLUSTER 2 IN TABLE I

x-coordinates (Number of neurons in the hidden layer)	y-coordinates (RMSE)
5	0.015936
7	0.014606
9	0.015315
10	0.01601
11	0.015648
14	0.015762
16	0.016232
19	0.016022
20	0.017614

TABLE III
VALUES OF NUMBER OF NEURONS IN THE HIDDEN LAYER AND THEIR CORRESPONDING VALUES OF RMSE WITHIN CLUSTER 2 WHICH IS SHOWN IN TABLE I

Number of neurons in the hidden layer	RMSE	Number of neurons in the hidden layer	RMSE
5	0.015502	13	0.015936
6	0.01536	14	0.016129
7	0.015269	15	0.016765
8	0.015459	16	0.015886
9	0.015686	17	0.016159
10	0.016048	18	0.016035
11	0.016591	19	0.016794
12	0.016305	20	0.016869

more than 10% of the initial range of 150 neurons. Thus the process of subdividing the range of values represented by the cluster marked as cluster 1 in Fig. 3 is performed.

Once this is done, guessing of the number of neurons in the hidden layer within the range covered by the cluster 1 in Fig. 3 is performed. Once the guesses and the corresponding errors are recorded, k-means++ clustering is implemented again on these guesses. The results shown in Table I show that cluster 2 is selected, since its RMSE value is the smallest among the cluster centroids. The range of values that are within cluster 2 shown in Table I, which are displayed in Table II, is now 10% of the original range of the number of neurons in the hidden layer of the BP network. Thus the clustering of this range of values by k-means++ clustering is not carried out.

The greedy algorithm now evaluates each value that is within the cluster 2 shown in Table I. The results of this evaluation, shown in Table III, reveal that the BP network needs to have 7 neurons in order to produce most accurate predictions. This is because the RMSE achieved by having 7 neurons in the hidden layer of the BP network is the smallest among the values shown in Table III.

TABLE IV
NUMBER OF THE HIDDEN NEURONS THAT PRODUCE SMALLEST RMSE ERRORS

Number of neurons in the hidden layer	RMSE
2	0.016672408
3	0.017079427
4	0.015702942
5	0.015283594
6	0.015360111
7	0.015105829
8	0.015355326
9	0.016196761
10	0.015987903

B. Results of running an exhaustive search of the BP network trained using Mackey-Glass time series without being optimized using the proposed KGA model

An exhaustive search for the optimal number of neurons in the hidden layer of the BP network is performed in order to verify that the proposed KGA model has indeed found the optimal architecture in predicting unknown values of the Mackey-Glass time series. To achieve this, the same standard BP network, which is not optimized by the proposed KGA model, is trained and simulated using the same Mackey-Glass time series data. For each run, the number of neurons in the hidden layer is increased from 1 neuron to 150 neurons. The RMSE error between the actual values of the out-of-sample data of the Mackey-Glass time series and the predicted values of the network is recorded for each run. The results of this verification, shown in Fig. 4, show that in general, having more than a neuron in the hidden layer of the BP network produces RMSE errors that are between 0.01 and 0.05. Fig. 4 also shows that having a single neuron in the hidden layer of the BP network produces a large RMSE which is close to 0.07. On the other hand, having between 2 and 10 neurons in the hidden layer would cause the BP network to produce the smallest RMSE errors. Table IV, which displays the RMSE errors that the BP network produces for having between 2 and 10 neurons in its hidden layer, shows that the minimum RMSE error is achieved when there are seven neurons in the hidden layer of the BP network. This value is the same value determined to be optimal by the proposed KGA model. This verifies the results of the optimization by the proposed KGA model, thus proving the effectiveness of the proposed KGA model in optimizing the BP network.

TABLE V
COMPARISON OF TIME TAKEN TO IMPLEMENT THE PROPOSED KGA MODEL AND IMPLEMENT EXHAUSTIVE SEARCH

Step	Average time per run (sec)	Time taken (sec)
Make 50 guesses x 30 runs per guess	35.26	52883.44
Perform k-means++ clustering of the 50 guesses	-	4.70
Make 17 guesses from the values within the cluster identified in step 3, with 30 runs each	29.24	14911.58
Perform k-means++ clustering of 17 guesses	-	1.09
Run greedy algorithm on remaining 16 values after clustering in step 4, with 30 runs each	22.19	10652.25
Total time taken to implement the proposed KGA model	-	**78453.06**
Total time taken to run an exhaustive search (150 guesses x 30 runs)	**33.33**	**150004.41**

TABLE VI
RMSE ERRORS OBTAINED BY USING THE OPTIMAL BP NETWORK AND OTHER METHODS CITED BY CHEN ET AL. [29]

Method	Prediction error (RMSE)
Auto-regressive model	0.19
Cascade correlation NN	0.06
Sixth-order polynomial	0.04
Linear prediction method	0.55
Genetic algorithm and fuzzy system	0.049
Product T-norm	0.0907
Optimal BP network	0.015

Fig. 4 also shows that when there are more than seven neurons in the hidden layer of the BP network, the prediction of the unknown values of the Mackey-Glass time series becomes less accurate. This is consistent with the findings of other authors found in the literature that the accuracy of the predictions of unknown values of the time series is reduced when the number of neurons in the hidden layer of the BP network increases.

C. Computational costs of running the proposed KGA model and the exhaustive search for the optimal number of neurons in the hidden layer of the BP network measured in real time

Table V compares the time taken to implement the proposed KGA model against the time taken perform an exhaustive search. From Table V we can see that the proposed KGA model only evaluates a total of 83 values of the number of neurons in the hidden layer of the BP network. On the other hand, a total of 150 values of the number of neurons in the hidden layer of the BP network have to be evaluated in order to find the optimal number of neurons in the hidden layer of the BP network. As a result, the time taken to complete the steps of the proposed KGA model shown in Table V is around 78453 seconds (≈21 hours). On the other hand, it takes around 150000 seconds (≈42 hours) to complete an exhaustive search for the optimal number of neurons in the hidden layer of the BP network. Thus the BP network user would spend around 50% less time to evaluate 55% values in order to find the optimal number of neurons in the hidden layer by implementing the proposed KGA model compared to an exhaustive search for the optimal number of neurons in the hidden layer of the BP network. This shows that the proposed KGA model is less computationally intensive compared to the exhaustive search of the number of neurons in the hidden layer of the BP network.

D. Comparison between the optimized BP network and other time series prediction methods

We can see from Table VI, which compares the results of using the optimal number of neurons in the hidden layer of the BP network against the results obtained using several methods found in reference [29], that the BP network with an optimal size of its hidden layer is inherently able to produce more accurate predictions compared to several other methods cited by reference [29] such as auto-regressive model, cascade-correlation NN, sixth-order polynomial and product T-norm. Since the task of the proposed KGA model is to find the optimal size of the hidden layer of the BP network, the RMSE errors shown in Table V indicate that the proposed KGA model is successful in optimizing the BP network which is able to produce more accurate predictions of a time series compared to several time series prediction methods proposed in the literature.

V. CONCLUSION AND FUTURE WORK

Evaluation on the proposed KGA model on the Mackey-Glass time series reveals that the proposed KGA model is able to find the optimal number of neurons in the hidden layer of the BP network. These findings correspond to the results of an exhaustive search in finding the optimal value of the number of neurons in the hidden layer of the same BP network without being optimized by the proposed KGA model, and also from the comparison of the performance of the optimized BP network with the performances of methods by other authors found in the literature.

In addition to that, the proposed KGA model is also able to evaluate fewer values in order to find the optimal number of neurons in the hidden layer of the BP network in a shorter time compared to an exhaustive search to achieve the same objective. Therefore we can say that the proposed KGA model is less computationally intensive than an exhaustive search for the number of neurons in the hidden layer of the BP network.

Based on these findings, one suggestion for future work on the proposed KGA model is to evaluate the performance of the proposed KGA model in optimizing BP networks that are being employed in problems other than prediction of time series, such as classification and regression tasks.

REFERENCES

[1] D. E. Rumelhart, G. E. Hinton and R. J. Williams, Learning Representations by Back-propagating Errors. *Nature*, vol. 323, pp. 533–536, Oct. 1986, doi:10.1038/323533a0

[2] R. Bustami, N. Bessaih, C. Bong and S. Suhaili. (2007, November). Artificial Neural Network for Precipitation and Water Level

Predictions of Bedup River. *IAENG Int. J. Comput. Sci*. [Online]. 34(2). Available: http://www.iaeng.org/IJCS/issues_v34/issue_2/IJCS_34_2_10.pdf

[3] O. A. Bahnusi and J. O. Pedro, Neural Network Based Identification and Approximate Predictive Control of a Servo-Hydraulic Vehicle Suspension System, *Eng. Lett.* 18(4). Available: http://www.engineeringletters.com/issues_v18/issue_4/EL_18_4_05.pdf

[4] J. A. Ramírez-Quintana, M. I. Chacon-Murguia and J. F. Chacon-Hinojos, Artificial Neural Image Processing Applications: A Survey. *Eng. Lett.* [Online]. 20(1). Available:http://www.engineeringletters.com/issues_v20/issue_1/EL_20_1_09.pdf

[5] H. Li and R. Kozma, A Dynamic Neural Network Method for Time Series Prediction using the KIII Model. In *Proc. Int. Jt. Conf. Neural Netw.*, vol. 1, Portland, Oregon. IEEE, Piscataway, USA, 2003, pp. 347-352. doi:10.1109/IJCNN.2003.1223370

[6] S. Kumar, *Neural Networks: A Classroom Approach*. New Delhi, India: Tata McGraw-Hill, 2004

[7] R. Reed, Pruning Algorithms – A Survey. *IEEE Trans. Neural Netw.*, vol. 4, pp. 740-747, Sept. 1993 doi:10.1109/72.248452

[8] E. Gonzalez-Romera, M. A. Jaramillo-Moran and D. Carmona-Fernandez, Monthly Electric Energy Demand Forecasting Based on Trend Extraction. *IEEE Trans. Power Syst.*, vol. 21, pp. 1946-1953, Nov. 2006 doi:10.1109/TPWRS.2006.883666

[9] D. S. Touretzky and D. A. Pomerleau, What's Hidden in the Hidden Layers? *Byte*, vol. 14, pp. 227-233, 1989

[10] G. Zhang, B. E. Patuwo and M. Y. Hu, Forecasting with Artificial Neural Networks: The State of the Art. *Int. J. Forecast.*, vol. 14, pp. 35-62, March 1998 doi:10.1016/S0169-2070(97)00044-7

[11] S. X. Yang, and N. Li, Power Demand Forecast based on Optimized Neural Networks by Improved Genetic Algorithm. In *2006 Int. Conf. Mach. Learn. Cybern.*, Dalian, China. IEEE, Piscataway, USA, pp. 2877-2881, Aug. 2006, doi:10.1109/ICMLC.2006.259073

[12] X. M. Gao, X. Z. Gao, J. M. A. Tanskanen and S. J. Ovaska, Power Prediction in Mobile Communication Systems Using an Optimal Neural-Network Structure. *IEEE Trans. Neural Netw.*, vol. 8, pp. 1446-1455, Nov. 1997, doi:10.1109/72.641467

[13] G. Castellano, A. M. Fanelli and M. Pelillo, An Iterative Pruning Algorithm for Feedforward Neural Networks. *IEEE Trans. Neural Netw.*, vol. 8, pp. 519-531 May 1997, doi: 10.1109/72.572092

[14] A. Yamazaki, T. B. Ludermir and M. C. P. de Souto, Global Optimization Methods for Designing and Training Neural Networks. In *Proc. VII Braz. Symp. Neural Netw. (SBRN 2002)*. IEEE, Piscataway, USA, pp 136-141, 2002 doi:10.1109/SBRN.2002.1181455

[15] F. van den Bergh and A. P. Engelbrecht, Cooperative Learning in Neural Networks using Particle Swarm Optimizers, *S. Afr. Comput. J.*, vol. 26, 84-90, 2000

[16] D. T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim and M. Zaidi, The Bees Algorithm, a Novel Tool for Complex Optimisation Problems. In *Proc. 2nd Virtual Int. Conf. Intell. Prod. Mach. Syst.* Elsevier, Oxford, pp. 454-459 (2006)

[17] W. Rebizant and D. Bejmert, Current-Transformer Saturation Detection with Genetically Optimized Neural Networks. *IEEE Trans. Power Deliv.*, vol. 22, pp. 820-827, April 2007, doi:10.1109/TPWRD.2007.893363

[18] H. Hyötyniemi, Structural Optimization of Feedforward Networks. In *STeP'96 - Genes, Nets and Symbols*, Vaasa, Finland. Finnish Artificial Intelligence Society (FAIS), pp. 25-34, 1996

[19] J. Lin, S. Chen, Y. Cao and H. Guan, Parameters Optimization on Dent around Fuel Filler of Auto Rear Fender Based on Intelligent Algorithm. GEC '09: In *Proc. first ACM/SIGEVO Summit Genet. Evol. Comput.*, Shanghai, China. ACM, New York, USA. pp. 321-328, 2009, doi:10.1145/1543834.1543878

[20] P. E. Black, P. E. (2005), Greedy algorithm. U.S. National Institute of Standards and Technology. [Online]. Available: http://www.itl.nist.gov/div897/sqg/dads/HTML/greedyalgo.html

[21] A. J. Hsieh, K. C. Fan and T. I. Fan, Handwritten Chinese Characters Recognition by Greedy Matching with Geometric Constraint. *Image Vis. Comput.*, vol. 14, pp. 91-104, March 1996, doi:10.1016/0262-8856(95)01043-2

[22] D. J. Williams and M. Shah, A Fast Algorithm for Active Contours and Curvature Estimation. *CVGIP: Image Understanding*, vol. 55, pp. 14-26, Jan 1992, doi:10.1016/1049-9660(92)90003-L

[23] J. Bang-Jensen, G. Gutin and A. Yeo, When the Greedy Algorithm Fails. *Discret. Optim.*, vol. 1, pp. 121–127, Nov. 2004, doi:10.1016/j.disopt.2004.03.007

[24] T. H. Cormen, C. H. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, Second Edition. Cambridge, Massachusetts: The MIT Press, 2001

[25] D. Arthur and S. Vassilvitskii, k-means++: the advantages of careful seeding. SODA '07 *Proc. Eighteenth Annu. ACM-SIAM Symp. Discret. Algoritm*. ACM, New York, USA. pp.1027-1035, 2007

[26] T. W. S. Chow and C. T. Leung, Neural Network based short-term load forecasting using weather compensation. *IEEE Trans. Power Syst.*, vol. 11, pp. 1736-1742, Nov. 1996, doi:10.1109/59.544636

[27] M. B. Yeary, Y. Zhai, T. Y. Yu, S. Nematifar and A. Shapiro, Spectral Signature Calculations and Target Tracking for Remote Sensing. *IEEE Trans. Instrum. Meas.*, vol. 55, 1430-1442, Aug. 2006 doi:10.1109/TIM.2006.876574

[28] M. C. Mackey and L. Glass, Oscillation and Chaos in Physiological Control Systems. *Science*, vol. 197, pp. 287-289, July 1977, doi:10.1126/science.267326

[29] Y. Chen, B. Yang and J. Dong, Time-series prediction using a local linear wavelet neural network, *Neurocomputing*, vol. 69, pp. 449-465 Jan 2006, doi:10.1016/j.neucom.2005.02.006

[30] I. Rojas, O. Valenzuela, F. Rojas, A. Guillen, L. J. Herrera, H. Pomares, L. Marquez and M. Pasadas, Soft-computing Techniques and ARMA Model for Time Series Prediction. *Neurocomputing*, vol. 71, pp. 519-537, January 2008, doi: 10.1016/j.neucom.2007.07.018

[31] Y. R. Park, T. J. Murray and C. Chen, Predicting Sun Spots Using a Layered Perceptron Neural Network. *IEEE Trans. Neural Netw.*, vol. 7, pp. 501-505, Mar 1996, doi:10.1109/72.485683