

## A PERFORMANCE EVALUATION OF PRUNING EFFECTS ON HYBRID NEURAL NETWORK

S. Y. Leow\*<sup>1</sup>, K. S. Yap<sup>1</sup>, H. J. Yap<sup>2</sup> and S. Y. Wong<sup>1</sup>

<sup>1</sup>College of Graduate Studies, Universiti Tenaga Nasional, 43000 Kajang, Selangor, Malaysia

<sup>2</sup>Department of Mechanical Engineering, Faculty of Engineering, University of Malaya, 50603 Kuala Lumpur, Malaysia

Published online: 05 October 2017

### ABSTRACT

In this paper, we explore the pruning effects on a hybrid mode sequential learning algorithm namely FuzzyARTMAP-prunable Radial Basis Function (FAM-PRBF) that utilizes Fuzzy ARTMAP to learn a training dataset and Radial Basis Function Network (RBFN) to perform regression and classification. The pruning algorithm is used to optimize the hidden layer of the RBFN. The experimental results show that FAM-PRBF has successfully reduced the complexity and computation time of the neural network.

**Keywords:** pruning; radial basis function network; fuzzy ARTMAP.

Author Correspondence, e-mail: [syleow88@hotmail.com](mailto:syleow88@hotmail.com)

doi: <http://dx.doi.org/10.4314/jfas.v9i4s.17>

### 1. INTRODUCTION

An artificial neural network (ANN) is generally constructed by three layers. They are input layer, hidden neuron layer and output layer. The inputs are collected and fed into the network through input layer. The second layer is the hidden neuron layer. This hidden neuron layers may have one or several hidden layers, which depends on the structure of the neural network. The output is commonly decided by an overall function of the ANN. The training data that fed to the network, training approach and weights of the neurons decide the overall function. Recently, hybrid models of ANNs have been proposed. Generally, there are three different methods in the hybridization of ANNs. They are model, algorithms and data [1].



### 1.1. Radial Basis Function Network

RBFN is a type of feedforward neural network that grounded on function approximation theory. This network searches the most outstanding match in multidimensional space within the knowledge that it learned from training data. RBF has good generalization, online learning capability and tolerant to noisy inputs [2]. These advantages make the RBF network widely applied in flexible control systems, dynamic systems and time-series prediction [3-7].

A simple RBFN consists of three layers. They are input layer, hidden neuron layer with  $Q$  hidden neurons and output layer. The connection between the input layer and hidden neuron layer is non-linear. However, the connection between the hidden neuron layer and output layer is linear. The hidden neurons implement radial basis functions. The  $y$  is the linear output of the RBFN, which is defined as follows:

$$y = f(\mathbf{X}) = \sum_{j=1}^Q \beta_j G_j(\mathbf{X}) \quad (1)$$

The  $\mathbf{X}$  is input vector,  $Q$  is the number of hidden neurons,  $\beta_j$  is the output weights and  $G_j(\mathbf{x})$  is radial basis function of the hidden neurons.

There are various forms of basis functions that can be applied to RBFN. In this paper, Gaussian function is selected. Gaussian function is commonly used. This function is local and only responses to the input that near the center. The absolute value of this function decreases continuously approaches zero in all dimensions when the inputs are away from its center. The  $G_j(\mathbf{X})$  is defined in Equation (2).

$$G_j(\mathbf{X}) = \exp\left(-\frac{\|\mathbf{X} - \mathbf{c}_j\|^2}{2\sigma_j^2}\right) \quad (2)$$

The  $\mathbf{c}_j$  is the center of the Gaussian function. The spread,  $\sigma$  controls the width of the Gaussian function. The  $\beta_j$  can be found by using  $\beta_j = \mathbf{G}_j^{-1} y$ . The  $\mathbf{G}_j$  may not be a square matrix. If the  $\mathbf{G}_j$  is not a square matrix,  $\beta_j$  can be found by using the pseudoinverse of  $\mathbf{G}_j$ ,

$$\beta_j = (\mathbf{G}^T \mathbf{G})^{-1} \mathbf{G}^T y \quad (3)$$

### 1.2. Fuzzy ARTMAP

Adaptive Resonance Theory (ART) has been developed by Carpenter and Grossberg in 1987 to overcome stability-plasticity dilemma. Fuzzy ARTMAP (FAM) is a supervised learning member of the ART family. FAM embeds fuzzy [21] logic operations to their neurons based on ART algorithm. The FAM is an incremental learning system with online learning

capability. FAM can preserve information learned from previous information (stable), while being flexible enough to learn new information incrementally (plastic). FAM does not forget the previous learned information, i.e., it does not have problem with catastrophic forgetting [8].

FAM has a pair of ART modules and a map field. They are ART-a and ART-b. Both ART modules have three neuron layers, i.e. normalization layer, input layer and recognition layer respectively. Training data that have inputs and target outputs are mandatory to be fed to FAM. Assume  $\{(X_1, T_1), (X_2, T_2), \dots, (X_i, T_i)\}$  denotes inputs and its target outputs. The  $i$ th training samples,  $X_i \in \mathbb{R}^M$  and  $T_i \in \mathbb{R}^L$ , are input vector with  $M$ -dimensions and target output vector with  $L$ -dimensions.  $F_0^a$  and  $F_0^b$  are normalization layers. In these layers, input vectors and its target output vectors are introduced to complement coding, i.e.,

$$\begin{aligned} A_i &= (\mathbf{X}_i, \mathbf{X}_i^c) = (\mathbf{X}_i, 1 - \mathbf{X}_i) \\ B_i &= (\mathbf{T}_i, \mathbf{T}_i^c) = (\mathbf{T}_i, 1 - \mathbf{T}_i) \end{aligned} \quad (4)$$

where  $A_i \in \mathbb{R}^{2M}$  is the complemented inputs vector, and  $B_i \in \mathbb{R}^{2L}$  is the complemented target outputs vector.  $F_1^a$  and  $F_1^b$  are input layers. These input layers connected in between normalization layers and recognition layers respectively. The vector that obtained from the  $A_i$  and  $B_i$  are presented to recognition layers via input layers.

The  $F_2^a$  and  $F_2^b$  are recognition layers. These two layers take complemented input and target output vectors separately. Every neuron encodes an input and its target outputs in these two layers. In the training phase, the number of hidden neurons can be increased. Each neuron  $j$  has its specific adaptive weights set in vector form. The weights with  $Q$  hidden neurons for both ART modules are  $w_j^a$  and  $w_j^b$ . The weights are imagined as hyper-rectangles. Meanwhile, the centers of both ART modules are  $c_j^a$  and  $c_j^b$ .

When a fresh hidden neuron is created (i.e.,  $Q \rightarrow Q+1$ ),  $w_j$  and  $c_j$  in both ART modules are fixed to be 1 and 0 respectively. The  $q$ , number of training sample which is assigned to neuron  $j$  in  $F_2^a$  layer, is set to be 0.

During the training phase, a normalized input vector is presented to ART-a. Meanwhile, its target output vector is presented to ART-b. An activation number is created to each neuron  $j$  in  $F_2^a$  by the choice function measurement. The hidden neuron which has the top activation number, is selected as the winning neuron. The winning neuron must accomplish winner-take-all competition rule.

$$T_j = \frac{|\mathbf{A}_i \wedge \mathbf{w}_j^a|}{r + |\mathbf{w}_j^a|} \quad (5)$$

The  $r$  is choice parameter and  $\mathbf{w}_j^a$  is weight vector of  $j$  neuron in  $F_2^a$ . The winning neuron is characterised as neuron  $J$ . After that, a vigilance test is performed. The winning neuron,  $\mathbf{w}_j^a$ , and  $\mathbf{A}_i$ , are matched the vigilance parameter,  $\rho_a$ ,

$$\frac{|\mathbf{A}_i \wedge \mathbf{w}_j^a|}{|\mathbf{A}_i|} \geq \rho_a \quad (6)$$

where  $\rho_a$  is  $[0,1]$ . If the winning neuron fails vigilance test, then a searching cycle of a new winning neuron is performed. This search cycle will be performed until the winning neuron fulfils the vigilance test. The similar procedure is presented in ART-b.

Once identify the winning neuron in the reorganization layer of ART-a, a vigilance test in ART-b is performed to its target output neuron in  $F_2^b$  (i.e.,  $\mathbf{w}_j^b$ ). This test is required to verify the neuron  $J$  is the concluding winner.

$$\frac{|\mathbf{B}_i \wedge \mathbf{w}_j^b|}{|\mathbf{B}_i|} \geq \rho_b \quad (7)$$

The  $\rho_b$  is a number between 0 and 1. If fails the vigilance test, a tracking match is presented as follows:

$$\rho_a = \frac{|\mathbf{A}_i \wedge \mathbf{w}_j^a|}{|\mathbf{A}_i|} + \sim \quad (8)$$

In  $F_2^a$ , to prevent the same neuron  $J$  to be chosen again, a very small positive constant number,  $\sim$  is added. In ART-a, a new search cycle starts with another threshold level of  $\rho_a$ .

If there is no neuron can fulfil Equation (7), a fresh neuron is created.

When the winner neuron  $J$  is examined, the centers and weights of winning neuron in recognition layers are updated as:

$$\mathbf{w}_J^a \leftarrow \mathbf{A}_i \wedge \mathbf{w}_J^a, \quad \mathbf{w}_J^b \leftarrow \mathbf{B}_i \wedge \mathbf{w}_J^b \quad (9)$$

$$q_J \leftarrow q_J + 1 \quad (10)$$

$$c_J^a \leftarrow \left(1 - \frac{1}{q_J}\right) c_J^a + \frac{\mathbf{A}_i}{q_J} \quad (11)$$

$$c_J^b \leftarrow \left(1 - \frac{1}{q_J}\right) c_J^b + \frac{\mathbf{B}_i}{q_J}$$

### 1.3. Pruning Algorithm

Pruning algorithm is not new to ANNs. Pruning algorithm has been proposed to improve generalization of an ANN. It can escape from overfitting problems in huge system [9]. A huge ANN carries unnecessary constraints. The network involves longer prediction responses, inessential knowledge storage and high-priced hardware implementation. The large ANN may overfit the training data and produce poor generalization performance. There are many types of pruning algorithms and can be divided into two groups [9]. One group prunes the hidden neurons based on the sensitivity of the error function. Another group is add a term to the pruning objective, for example, the hidden neurons with small weights can be pruned.

Pruning algorithm can be applied to an ANN during training procedure or after each training epoch. A resource allocating network (RAN) which was firstly introduced by Platt in 1991 is used to allocate hidden neurons and prune insignificant hidden neurons [10]. RAN was further enhanced. Minimal resource allocating network (MRAN) can grow and prune neurons. Pruning strategy can be applied to an ANN during training procedure or after each training epoch. In radial basis function networks with dynamic decay adjustment (RBFN-DDA) algorithm, the pruning strategy is applied to the network after each training epoch [11]. Besides, in [12-13] introduced growing and pruning RBF (GAP-RBF) and generalized growing and pruning RBF (GGAP-RBF). A similar algorithm, FAMDDA with temporary neurons (FAMDDA-T) that consists dynamic decay adjustment [23] and pruning strategy is developed by [14]. Recently, in [15] has proposed a self-organizing recurrent radial basis function (SR-RBF) neural network that designed based on spiking mechanism and improved Levenberg-Marquardt (LM) algorithm. The hidden neuron layer in this algorithm can add or prune hidden neurons by calculating the spiking strength between hidden and output neurons in recurrent radial basis function neural network.

## 2. METHODOLOGY

This paper proposes an algorithm that ties the pruning of insignificant hidden neurons on the relevance of the hidden neurons in a hybrid neural network. This FAM-PRBF algorithm is using sorting methods to measure the relevance of each hidden node in the proposed algorithm. A simple pruning technique is applied to remove the insignificant hidden neurons in RBFN.

Fig. 1 illustrates the architecture of the new algorithm. A remarkable arrangement of the proposed algorithm the center of RBFN hidden neurons are determined by the hidden

neurons in FAM. The center of hidden neurons in a RBFN are selected randomly from the training dataset. A dataset is required to be divided into training dataset and testing dataset. The training dataset is presented to FAM. The task of FAM in this hybrid neural network is to learn from the training dataset.

During the training process, the FAM hidden neurons grow as requested by the training dataset. When the training process completed, the center weights of FAM,  $c_j^a$  are employed to compute the hidden neurons in RBFN. The  $c_j^a$  is substituted as center of Gaussian function in Equation (2) to compute the activation function of RBFN. The spread,  $\sigma_j$  is a tuneable parameter that has to be determined before training phase. Then, the output weight,  $w_j$  are computed.

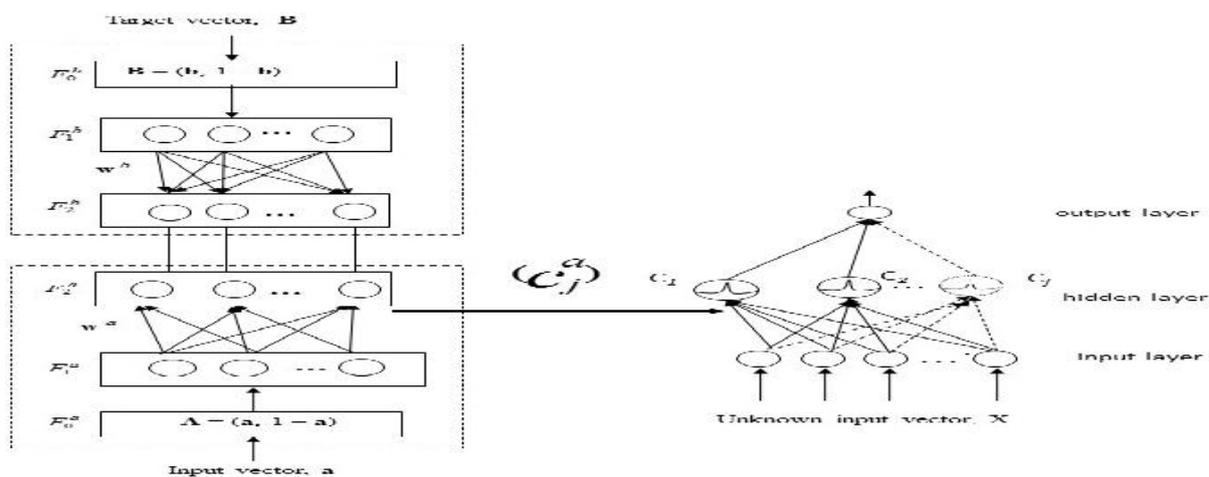
After computing the RBFN, testing dataset is applied to the network to test the accuracy of this hybrid neural network [22]. Next, the pruning algorithm is implemented in the RBF hidden layer based on the relevance of the hidden neurons. The pruning threshold,  $\theta$  which decides the insignificant hidden neurons, is defined before the pruning algorithm is executed. The number of insignificant hidden neurons to be pruned which is defined in percentage form has to be converted to decimal form within the range (0,1], then multiplied with the total number of hidden neurons,  $Q$ , to set the  $\theta$ .

The pruning process of the proposed algorithm is described as follow:

Step 1: Re-allocate the absolute value of output weight,  $w_j$  that are arranged firstly in RBFN of FAM-PRBF network in descending order. The hidden neurons are re-allocated together with the  $w_j$ .

Step 2: If the  $q$ th hidden neurons is greater than  $\theta$ , it's output weight,  $w_j$  will be remain and store. The  $q$ th hidden neuron is defined as insignificant hidden neuron, if it is re-allocated lower than  $\theta$ . The  $w_j$  of the insignificant neurons will be set to 0. The connections that linked to the insignificant hidden neurons will be trimmed.

$$S_j = \begin{cases} S_j, q > \theta \\ 0, q \leq \theta \end{cases} \quad (12)$$



**Fig.1.** Architecture of FAM-PRBF network

Step 3: Re-organize the  $c_j$  and re-adjust using Equation (1), (2) and (3). Then, update the RBFN of FAM-PRBF.

After pruning process, the training and testing datasets are re-applied to the RBFN of FAM-PRBF. The training and testing performances of the RBFN are re-measured.

### 3. RESULTS AND DISCUSSION

In this study, the performance of the pruning effects on FAM-PRBF is evaluated on four benchmarking problems (two classification dataset and two regression dataset) from UCI machine-learning repository [16]. The two multi-class classification problems that have been considered are Image Segment and Satellite Image. However, the two regression problems that have been studied are Abalone and MPG. All the experiments were performed in MATLAB 7.11 running on a Core i5, 2.6GHz CPU with 4G RAM surroundings.

The specification of the experiments are presented in Table 1. Each dataset was distributed into training and testing data as presented in Table 1. All the outputs are normalized into the range [0, 1] in the experiments. The inputs are normalized into the range [-1, 1] in classification problems. The inputs are normalized into the range [0, 1] in regression problems. The spread of the Gaussian function in RBFN,  $\sigma$  and vigilance parameters,  $\rho_a$  and  $\rho_b$  of FAM had to be determined before conducting the experiments. The  $\rho_a$  is set to 1 and  $\rho_b$  has to be tuned the in classification experiments. However,  $\rho_b$  is set to be 0 and  $\rho_a$  has to be tuned in regression experiments.

**Table 1.** Specification of benchmark datasets for experiments

Dataset	Attributes	Class	Training Data	Testing Data
Abalone	8	n/a	3,000	1,177
MPG	7	n/a	320	72
Image Segment	19	7	1500	810
Satellite Image	36	6	4435	2000

The objective of Abalone experiment is to predict the age of abalone based on their physical dimensions. The intention of MPG experiment is to predict the fuel consumption of numerous kinds of cars in term of miles per gallon. Besides, the main objective of image segmentation experiment is to classify each region into one of the seven outdoor images through 19 attributes. The seven outdoor images are brick facing, sky, foliage, cement, window, pathway and grass. Each image is a 3X3 region. The main goal of satellite image experiment is to categorize each region into one of the six categories namely very dump grey soil, damp grey soil, cotton crop, soil with vegetation stubble, red soil and grey soil by using 36 attributes.

The average training and testing accuracy are collected based on 50 trials. These accuracies are indicated in root mean square (RMS) error for regression problems and in accuracy rates (%) for classification problems.

### 3.1. Comparisons on Performance Evaluations

The performance of FAM-PRBF (before pruning) on regression problems is reported in Table 2. This performance is compared with other learning algorithms. The performances of the compared learning algorithms in both regression and classification experiments are straight denoted from the literature.

**Table 2.** Results of the regression experiments before pruning

Dataset	Algorithms	RMSE		Number of Hidden Neurons
		Training	Testing	
Abalone	FAM-RBF	0.0742	0.0773	72
	GAP-RBF	0.0963	0.0966	23.62
	OS-ELM (RBF) [17]	0.0759	0.0783	25
	MRAN [17]	0.0836	0.0837	87.571
MPG	FAM-RBF	0.0560	0.0744	64
	GAP-RBF	0.1144	0.1404	3.12
	OS-ELM (RBF) [17]	0.0696	0.0759	25
	MRAN [17]	0.1086	0.1376	4.46

The number of hidden neurons in FAM-PRBF (before pruning) for Abalone experiment is similar with MRAN. However, the training and testing RMS errors of FAM-PRBF (before pruning), 0.0742 and 0.0773 in Abalone experiment are significantly better than the training and testing RMS errors of MRAN (0.0836 and 0.0827). In this experiment, the training and testing RMSE values of the FAM-PRBF (before pruning) are noticeable better than other algorithms. Before the pruning algorithm in FAM-PRBF is performed in MPG experiment, the number of hidden neurons, 64 is slightly larger than the number of hidden neurons in the compared algorithms. The training and testing RMSE values of FAM-PRBF (before pruning) are better than the compared learning algorithms. The FAM-PRBF (before pruning) has comparable training and testing RMS errors (0.0560 and 0.0744) with the training and testing RMS errors of OS-ELM (RBF) (0.0696 and 0.0759).

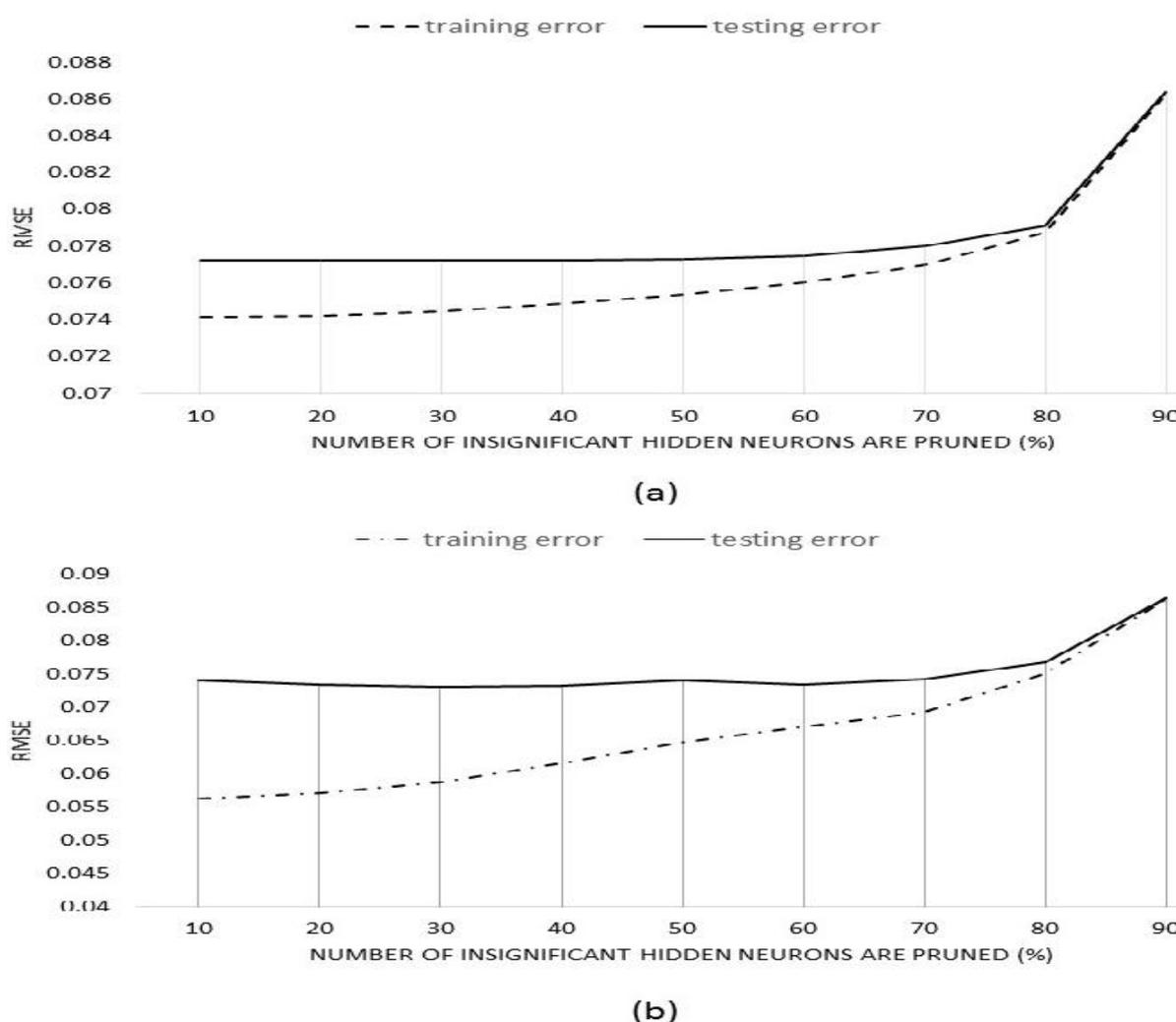
Fig. 2 summarizes the training and testing RMS errors on different numbers of insignificant hidden neurons are pruned (%) for Abalone and MPG experiments correspondingly. With regards to the Abalone experiment, both RMS errors and number of hidden neurons (after pruning) are close to the training and testing RMS error of OS-ELM (0.0759 and 0.0783) and hidden neurons, 25. In comparison with GAP-RBF that has 23.62 hidden neurons, GAP-RBF obtains higher training and testing RMS errors at 0.0963 and 0.0966. However, the training RMS error of the FAM-PRBF (after pruning) in MPG experiment increases with number of insignificant hidden neurons are pruned. When 10% to 70% insignificant hidden neurons are pruned, the testing RMS error steadily increases. When there are 70% of insignificant hidden neurons are pruned, the training and testing RMS error are 0.0692 and 0.0742, and the number of hidden neuron in the new RBFN is 19.24. This result is comparable with the OSELM (RBF) network which has 25 hidden neurons, obtains training and testing RMS error at 0.0696 and 0.759.

As observed from Table 3, FAM-PRBF (before pruning) obtains a comparable performance results in both classification experiments. The training and testing performances of FAM-PRBF (before pruning) in both experiments are better than the compared algorithms. However, the number of hidden neurons of FAM-PRBF (before pruning) in both experiments are visibly larger than the compared learning algorithms.

The training and testing accuracies on different numbers of insignificant hidden neurons are pruned (%) for Image Segmentation and Satellite Image experiments respectively are tabulated in Fig. 3. The testing accuracy of the FAM-PRBF decreases steadily in Satellite Image experiments when 10% to 60% of insignificant hidden neurons are pruned. When 60%

of the insignificant hidden neurons are removed in Satellite Image experiment (453.2 hidden neurons in FAM-PRBF), the training and testing accuracies are better than the training and testing accuracies of OS-ELM (RBF).

Based on the performance evaluations in Fig. 2 and 3, it can be noticed that there is a sharp falling on the testing accuracy or raising on the testing RMS error when there are about 80% of insignificant hidden neurons are pruned. Further pruning (prune more than 80% of insignificant hidden neurons) worsens training and testing performances of the network. This is a natural stopping point of pruning strategy in this algorithm. The pruning strategy has successfully remove the insignificant hidden neurons in this proposed algorithm.

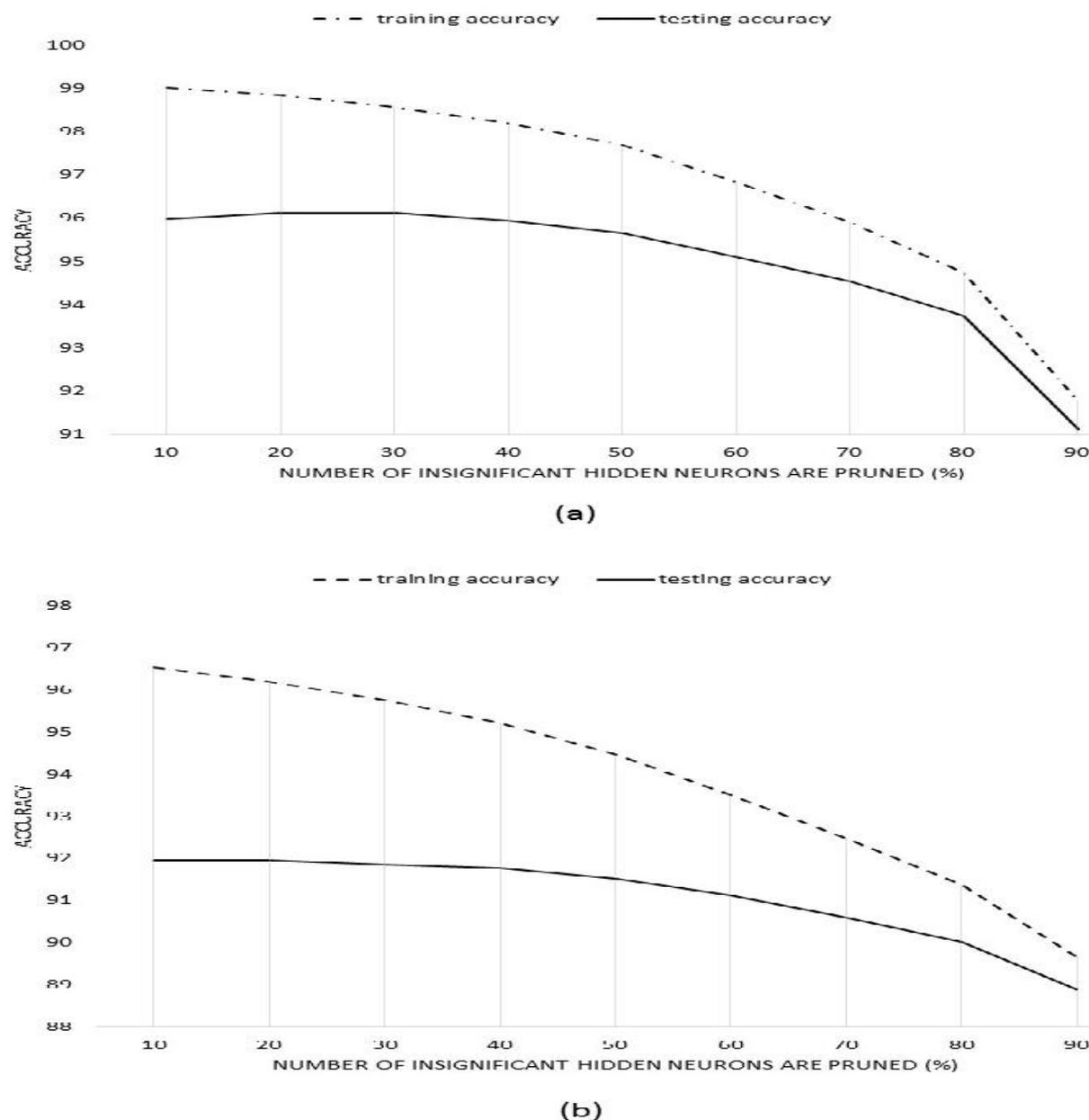


**Fig.2.** Training and testing RMS errors for (a) Abalone dataset (b) MPG dataset

**Table 3.** Results of the classification experiments before pruning

Dataset	Algorithms	Accuracy (%)		Number of Hidden Neurons (Before Pruning)
		Training	Testing	
Image Segmentation	FAM-RBF	99.16	95.71	413.8
	FAM-OELM [18]	96.39	95.06	200
	OS-ELM (RBF) [17]	96.65	94.53	180
	FAM [19]	n/a	93.09	33
	MRAN [20]	n/a	93.30	53
	FAM-RBF	96.80	92.12	1133
Satellite Image	FAM-OELM [18]	91.36	91.10	390
	OS-ELM(RBF) [17]	93.18	89.01	400
	FAM [19]	n/a	82.5	69
	MRAN [20]	n/a	86.36	20

We can conclude that the significant hidden neurons play important participations to this algorithm. The participation is defined by the absolute value of output weight,  $w_j$  in RBFN. The output weight that is nearly zero has less influence to the RBFN. Therefore, it can be pruned. The training error is expected to be high in regression experiments and the training accuracy is expected to be felled in classification experiments when pruning algorithm is applied to FAM-PRBF network. During the training phase (before pruning), training data has been used to estimate the number of hidden neurons in this proposed algorithm. The hidden neurons are created as required by the training data. When pruning algorithm is applied, the training error raises because the FAM-PRBF network becomes underfit after pruning.



**Fig.3.** Training and testing accuracies for (a) Image Segmentation dataset (b) Satellite Image dataset

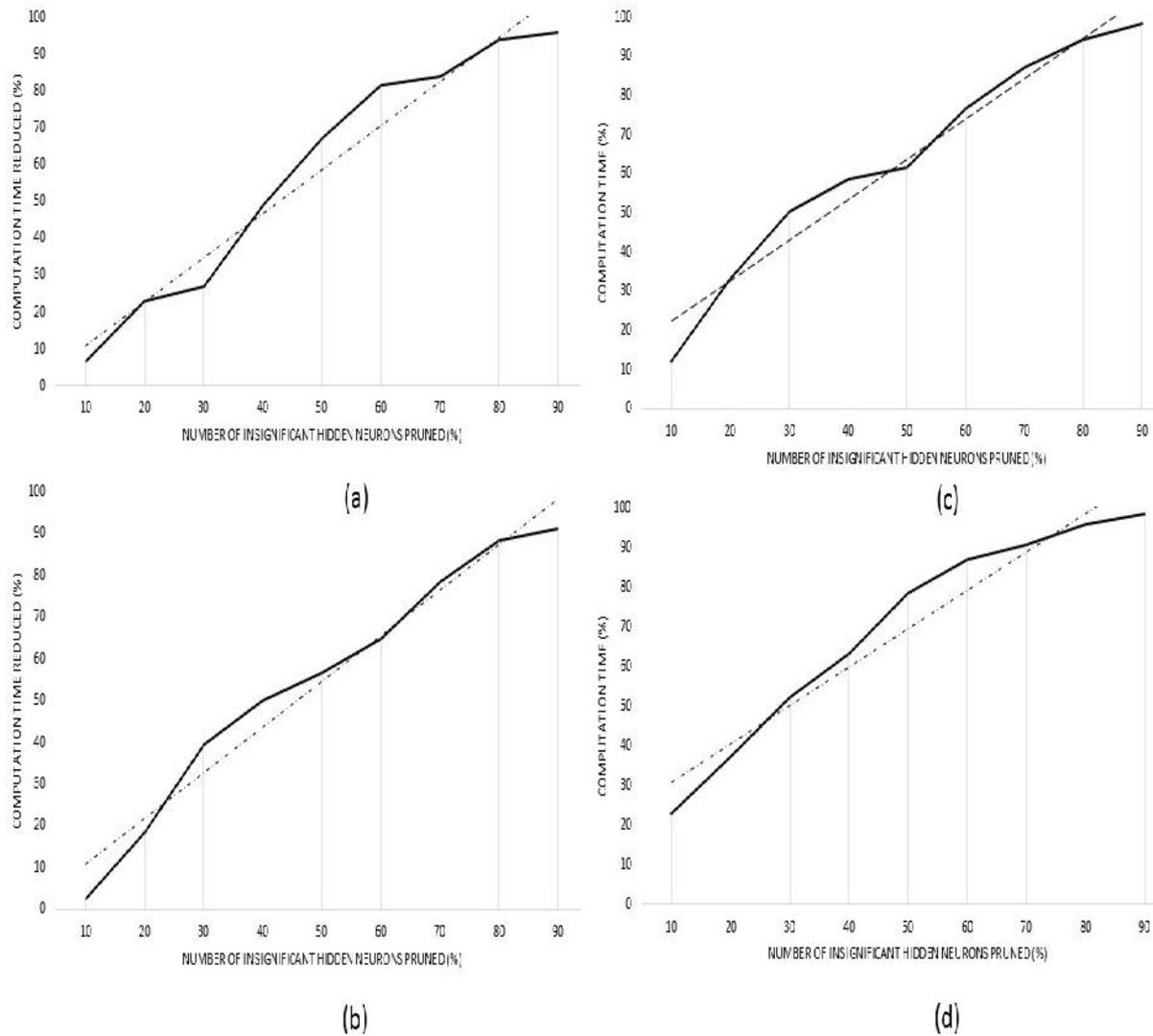
Some of the insignificant hidden neurons which are related and important to training dataset were removed in the pruning phase.

In both regression experiments, the best pruning results occurs when there are 30% of insignificant hidden neurons is pruned. The testing errors in both experiments at this pruning threshold were improved. When 20% to 30% of insignificant hidden neurons are removed, the testing error in Abalone and MPG experiments was felled from 0.077265 to 0.077224 and 0.073469 to 0.073083 respectively. The testing accuracy in Image Segmentation experiment is slightly improved from 95.97% to 96.11% when 10% to 30% insignificant hidden neurons

are pruned. This pruning threshold has solved the overfitting problems and improved the testing performances in these experiments. The repetitive hidden neurons that detract the testing performance of the network has been recognized and removed. Therefore, in the previous study, [5], the pruning threshold was set at 30% for the regression experiments.

### **3.2. Comparisons on Computation Time**

Fig. 4 reports the computation time reduced of RBFN (after pruning) in these four experiments. The ideal computation time reduced that is expected to be linearly proportional to the number of pruned insignificant hidden neurons is represented in dashed line in these four experiments. However, the solid line denotes the real-world computation time of the new RBFN in the proposed algorithm.



**Fig.4.** Computation time reduced (%) versus number of insignificant hidden neurons pruned (%) for: (a) Abalone dataset, (b) MPG dataset, (c) Image segmentation dataset and (d) Satellite image dataset

The computation time of RBFN reduced is calculated based on percentage change formula below:

$$C.T. \text{ reduced } (\%) = \frac{\text{before} - \text{after}}{\text{before}} \times 100 \% \tag{13}$$

The C.T. Reduced indicates the computation time reduced in percentage form, before indicates the computation time before pruning in seconds, and after indicates the computation time after pruning in seconds.

These four graphs have successfully reported the relationship between the numbers of pruned insignificant hidden neurons and the computation time of the RBFN (after pruning) is linearly proportional.

#### 4. CONCLUSION

In conclusion, the pruning algorithm has been successfully proposed in FAM-PRBF. The pruning algorithm has reduced the size of hidden neurons layer in the RBFN of FAM-PRBF network and computation time of FAM-PRBF network. The definition of pruning algorithm in this paper is to remove the insignificant hidden neurons that has output weights nearly zero and less influence to the network. The performances of FAM-PRBF on regression and classification benchmarking problems are compared with other well-known learning algorithms. Using the correct pruning threshold, FAM-PRBF can return substantial or better training and testing accuracies. The pruning algorithm in FAM-PRBF is an exciting topic for further research.

#### 5. REFERENCES

- [1] Gutiérrez P A, Hervás-Martínez C. Hybrid artificial neural networks: models, algorithms and data. In International Work-Conference on Artificial Neural Networks, 2011, pp. 177-184
- [2] Poggio T, Girosi F. Networks for approximation and learning. Proceedings of the IEEE, 1990, 78(9):1481-1497
- [3] Lu Y, Zhang T, Zeng Z, Loo J. An improved RBF neural network for short-term load forecast in smart grids. In IEEE International Conference on Communication Systems, 2016, pp. 1-6
- [4] Jain L C, Seera M, Lim C P, Balasubramaniam P. A review of online learning in supervised neural networks. Neural Computing and Applications, 2014, 25(3-4):491-509
- [5] Leow S Y, Wong S Y, Yap K S, Yap H J. A new hybrid fuzzy ARTMAP and radial basis function neural network with online pruning strategy. In 7th IEEE Control and System Graduate Research Colloquium, 2016, pp. 17-21
- [6] Baghaee H R, Mirsalim M, Gharehpetan G B, Talebi H A. Nonlinear load sharing and voltage compensation of microgrids based on harmonic power-flow calculations using radial basis function neural networks. IEEE Systems Journal, 2017, 99:1-11
- [7] Dong F, Lei X, Chou W. A dynamic model and control method for a two-axis inertially stabilized platform. IEEE Transactions on Industrial Electronics, 2017, 64(1):432-439
- [8] Carpenter G A, Grossberg S, Markuzon N, Reynolds J H, Rosen D B. Fuzzy ARTMAP: A

---

neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, 1992, 3(5):698-713

[9] Reed R. Pruning algorithms-A survey. *IEEE Transactions on Neural Networks*, 1993, 4(5):740-747

[10] Platt J. A resource-allocating network for function interpolation. *Neural Computation*, 1991, 3(2):213-225

[11] Paetz J. Reducing the number of neurons in radial basis function networks with dynamic decay adjustment. *Neurocomputing*, 2004, 62:79-91

[12] Huang G B, Saratchandran P, Sundararajan N. An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 2004, 34(6):2284-2292

[13] Huang G B, Saratchandran P, Sundararajan N. A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. *IEEE Transactions on Neural Networks*, 2005, 16(1):57-67

[14] Tan S C, Rao M V, Lim C P. On the reduction of complexity in the architecture of fuzzy ARTMAP with dynamic decay adjustment. *Neurocomputing*, 2006, 69(16):2456-2460

[15] Lu C, Han H, Qiao J, Yang C. Design of a self-organizing recurrent RBF neural network based on spiking mechanism. In *35th IEEE Chinese Control Conference*, 2016, pp. 3624-3629

[16] Blake C, Merz C J. *UCI repository of machine learning databases*. Irvine: University of California, 1998

[17] Liang N Y, Huang G B, Saratchandran P, Sundararajan N. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on Neural Networks*, 2006, 17(6):1411-1423

[18] Wong S Y, Yap K S, Yap H J, Tan S C. A truly online learning algorithm using hybrid fuzzy ARTMAP and online extreme learning machine for pattern classification. *Neural Processing Letters*, 2015, 42(3):585-602

[19] Tan S C, Rao M V, Lim C P. An online pruning strategy for supervised ARTMAP-based neural networks. *Neural Computing and Applications*, 2009, 18(4):387-395

[20] Li G, Liu M, Dong M. A new online learning algorithm for structure-adjustable extreme

learning machine. *Computers and Mathematics with Applications*, 2010, 60(3):377-389

[21] Tay K M, Lim C P. On the use of fuzzy inference techniques in assessment models: Part II: Industrial applications. *Fuzzy Optimization and Decision Making*, 2008, 7(3):283-302

[22] Yap K S, Lim C P, Abidin I Z. A hybrid ART-GRNN online learning neural network with a  $\lambda$ -insensitive loss function. *IEEE Transactions on Neural Networks*, 2008, 19(9):1641-1646

[23] Paetz J. Reducing the number of neurons in radial basis function networks with dynamic decay adjustment. *Neurocomputing*, 2004, 62:79-91

**How to cite this article:**

Leow SY, Yap KS, Yap HJ, Wong SY. A performance evaluation of pruning effects on hybrid neural network. *J. Fundam. Appl. Sci.*, 2017, 9(4S), 293-309.