Brief Papers

A Hybrid ART-GRNN Online Learning Neural Network With a ε-Insensitive Loss Function

Keem Siah Yap, Chee Peng Lim, and Izham Zainal Abidin

***Abstract*—In this brief, a new neural network model called generalized adaptive resonance theory (GART) is introduced. GART is a hybrid model that comprises a modified Gaussian adaptive resonance theory (MGA) and the generalized regression neural network (GRNN). It is an enhanced version of the GRNN, which preserves the online learning properties of adaptive resonance theory (ART). A series of empirical studies to assess the effectiveness of GART in classification, regression, and time series prediction tasks is conducted. The results demonstrate that GART is able to produce good performances as compared with those of other methods, including the online sequential extreme learning machine (OSELM) and sequential learning radial basis function (RBF) neural network models.**

***Index Terms*—Adaptive resonance theory (ART), Bayesian theorem, generalized regression neural network (GRNN), online sequential extreme learning machine.**

I. Introduction

Over the past two decades, many different neural network models have been developed for pattern recognition. In general, both multilayer perceptron (MLP) [1], [2] and radial basis function (RBF) [1], [3] networks have shown good results in a variety of pattern recognition problems. The MLP network tackles the pattern recognition problem by constructing a nonlinear transformation of combined sigmoid functions of its hidden neurons. On the other hand, the RBF network tackles the problem by combining nonlinear semiparametric functions such as the Gaussian kernel function. However, the number of hidden neurons of MLP and the number of kernel functions of RBF have to be predetermined by a series of cross-validation sequences, or by trial-and-error, which could be time consuming. Apart from that, the training process is time consuming because the training data set has to be presented to the networks repeatedly during the training phase.

A search in the literature reveals that sequential learning models based on RBF have become a popular research area for pattern recognition. These include the resource allocation network (RAN) [4] and its extensions, i.e., resource allocation network with extended Kalman filter (RANEKF) [5], minimum resource allocation network (MRAN) [6], growing and pruning RBF (GAPRBF) [7], and generalized growing and pruning RBF (GGAPRBF) [8]. Recently, an online learning model known as the online sequential extreme learning machine (OSELM) is proposed [9]. OSELM is developed based on a batch-mode version of ELM [10]. It uses the sequential least squares method to minimize the error function. OSELM works with both additive-sigmoid and RBF hidden neurons. For additive-sigmoid hidden neurons, the input weights and biases are randomly generated (and remain unchanged during the training phase), and the output weights are analytically determined. Similarly, for OSELM with RBF hidden neurons, the centers and widths of the RBF functions (input weights) are randomly generated, and the output weights are analytically determined.

In this brief, a hybrid online learning neural network called the generalized adaptive resonance theory (GART) model is proposed. This hybrid network first uses a modified Gaussian adaptive resonance theory (GA) model [11] to perform unsupervised clustering by compressing the training samples into several categories. It then employs an enhanced generalized regression neural network (GRNN) [12] to construct the decision function for prediction. This hybrid network is able to handle both classification and regression problems with an online learning capability. The motivations behind the development of GART are as follows:

1) to embed the decision function of the GRNN into the modified GA (MGA) model;
2) to reduce the number of kernels in the GRNN by compressing the training samples using MGA;
3) to perform "online learning" right from the first sample. Here, online learning is defined as follows:
   a) at any time of the training cycle, only the newly arrived sample is needed, instead of all past samples [9], for learning;
   b) ability to conduct one-pass learning through all training samples, with no reiteration through the training set;
   c) ability to conduct incremental and online learning of new knowledge without the disturbing the existing knowledge base;
   d) ability to predict the target output for a given (unlabeled) input sample at any time during the training cycle.

This brief is organized as follows. Section II describes the learning algorithm of GART. Section III presents a total of nine experimental studies and comparison of results for GART, OSELM, and other methods. A summary of this brief, with suggestions for further work, is presented in Section IV.

II. GART Learning Algorithm

The GRNN is a memory-based supervised learning neural network proposed by Specht [12]. It is able to provide an estimate of continuous variables and to converge to the underlying linear and/or nonlinear regression surface. The GRNN learning process is instantaneous as it is able to perform "one-pass" learning through the training samples. All neurons in the GRNN are treated equally as kernels, and are use to compute the independent probability density function (pdf), respectively, given a new input sample. If there is a new input sample that needs to be learned, the GRNN creates a new neuron to memorize the new sample. Generalization is achieved by using an estimator that comprises combinatory pdfs, in accordance with the Parzen-window probability density estimation method [13].

On the other hand, the unsupervised Gaussian ART (GA) [11] model assumes a learning algorithm based on a hybrid Gaussian classifier and ART [14], [15]. The learning objective of GA is to compress a new training sample into one of the existing categories, based on the pdf and Bayesian theorem. If all existing categories fail to accommodate

Manuscript received March 9, 2007; revised November 16, 2007 and February 27, 2008; accepted April 30, 2008. First published July 15, 2008; current version published September 4, 2008.

K. S. Yap and I. Z. Abidin are with College of Engineering, Universiti Tenaga Nasional, Selangor 43009, Malaysia (e-mail: yapkeem@uniten.edu.my; Izham@uniten.edu.my).

C. P. Lim is with School of Electrical and Electronic Engineering, University of Science Malaysia, Penang 14300, Malaysia (e-mail: cplim@eng,usm.my).

Digital Object Identifier 10.1109/TNN.2008.2000992

the new sample, a new category is created to represent the sample. A supervised GA model, i.e., Gaussian ARTMAP (GAM) [11], is also proposed for pattern classification. However, both GA and GAM cannot handle regression problems.

By exploiting the advantages of GA and the GRNN, in this brief, we propose a GART model. During the training phase, two MGA modules are created, i.e., MGA-a and MGA-b. MGA-a is used to identify the centers and standard deviations of the input samples while MGA-b is used to form the desired outputs (kernel labels) and its standard deviations. During the prediction phase, an enhanced GRNN is used to predict the output given a new input sample.

Similar to GA and the GRNN, GART is capable of performing online learning by adjusting its weights (centers and standard deviations of a kernel and its label) or by creating a new kernel and its label. These processes are fast and with a low computational cost. Note that, in this brief, GART is an improved version of the hybrid model presented in [16] and [17]. While the standard quadratic loss function is used in [16] and [17], a ε-insensitive loss function (which becomes the Laplacian loss function when $\varepsilon = 0$) is adopted in GART. Instead of the Gaussian function in [16] and [17], GART uses the exponential kernel function for formulating the choice and match functions of ART. The learning equation of the standard deviation of kernels is based on Laplacian distribution, rather than Gaussian distribution as in [16] and [17]. Furthermore, there is no match tracking mechanism in GART. Other improvements in this work, as compared with [16] and [17], include extensive empirical studies on classification, regression, and time series prediction problems; statistical quantification of the results by the bootstrap technique; and performance comparison with other sequential learning models.

In general, the dynamics of GART are as follows.

A. Training

The training samples presented to MGA-a and MGA-b are $\{(\mathbf{A}_1, \mathbf{B}_1), (\mathbf{A}_2, \mathbf{B}_2), \ldots, (\mathbf{A}_k, \mathbf{B}_k) \cdots\}$, where $\mathbf{A}_k \in \boldsymbol{R}^M$ and $\mathbf{B}_k \in \boldsymbol{R}^1$ are the input vector and kernel label of kth training sample, respectively. Note that in the following discussion, the equations and variables are based on MGA-a with input sample $\mathbf{A}_k$. Equations and variables of MGA-b with input kernel label $\mathbf{B}_k$ are the same but with subscript "b" instead of "a."

B. Competition

The input sample is presented to MGA-a, with its kernel label to MGA-b, for computing the choice and match functions. The choice and match functions are defined based on the Bayesian theorem. The Bayesian posterior probability for category-j of MGA-a to input sample $\mathbf{A}_k$ is

$$P(j|\mathbf{A}_k) = \frac{P(\mathbf{A}_k|j)P(j)}{P(\mathbf{A}_k)}. \tag{1}$$

The prior probability is

$$P(j) = \frac{n_j^a}{\sum_{i=1}^{N} n_i^a} \tag{2}$$

where N is the total number of output classes and n_i is the number of samples of category-i. The pdf is

$$P(\mathbf{A}_k) = \sum_{i=1}^{N} P(\mathbf{A}_k|i)P(i) \tag{3}$$

where $P(\mathbf{A}_k|j)$ is a kernel function in the exponential form [18], [19] that is used to measure the similarity between $\mathbf{A}_k$ and category-j, and is defined as

$$P(\mathbf{A}_k|j) \propto \exp\left(-\frac{1}{2}\lambda(\delta_{\mathrm{kj}}^a)\right) \tag{4}$$

where $\lambda(.)$ is the loss function.

While the original GA model uses a standard quadratic loss function, a ε -insensitive loss function is used in GART, as follows:

$$\lambda(\delta_{kj}^a) = \begin{cases} 0, & \delta_{kj}^a \le \varepsilon_a \\ \delta_{\mathrm{kj}}^a - \varepsilon_a, & \text{otherwise} \end{cases} \tag{5}$$

where $\varepsilon_a \ge 0$ is a predefined parameter, and

$$\delta_{kj}^a = \sum_{i=1}^{M} \left| \frac{\mu_{ji}^a - \mathbf{A}_{\mathrm{ki}}}{\sigma_{ji}^a} \right| \tag{6}$$

with $\boldsymbol{\mu}_j^a, \boldsymbol{\sigma}_j^a$, and n_j^a, respectively, the center, standard deviation, and count of category-j. The ε-insensitive loss function is used because the quadratic loss function can be influenced by outliers especially in the case of noisy data, which may lead to inaccurate recognition. In addition, the ε-insensitive loss function incurs no penalty to small noise values in the training samples [18].

The choice and match functions of MGA-a are defined by (1) and (4), respectively. The winning category of MGA-a, denoted as category-J, is the one with the highest choice function, with the condition that both match functions of MGA-a and MGA-b are larger than or equal to their respective vigilance parameters, i.e.,

$$J = \arg\max_j (P(j|\mathbf{A}_k)) \tag{7}$$

subject to $P(\mathbf{A}_k|J) \ge \rho_a$ and $P(B_k|J) \ge \rho_b$, where ρ_a and ρ_b are predefined vigilance parameters of MGA-a and MGA-b between 0 and 1. For simplicity, ρ_b is set to 1 for classification problems, and to the same value as ρ_a for regression problems. Note that the procedure used to select a winner in MGA-a is simpler as compared with that of original GAM, e.g., the match tracking mechanism is omitted. Besides, $P(\mathbf{A}_k)$ does not affect the selection of the winner as it is the same for all categories.

C. Learning

Learning involves the adjustment of the center, standard deviation, and counts of MGA-a with the following:

$$n_J^a \leftarrow n_J^a + 1$$
$$\boldsymbol{\mu}_J^a \leftarrow \left(1 - \frac{1}{n_J^a}\right)\boldsymbol{\mu}_J^a + \frac{\mathbf{A}_k}{n_J^a}$$

and

$$\boldsymbol{\sigma}_J^a \leftarrow \left(1 - \frac{1}{n_J^a}\right)\boldsymbol{\sigma}_J^a + \left|\frac{\boldsymbol{\mu}_J^a - \mathbf{A}_k}{n_J^a}\right|. \tag{8}$$

These equations are different from those in [11], as a different type of loss function is used in GART.

D. Addition of New Category

If none of the existing categories is able to satisfy (7), a new category is created to represent the new sample, i.e.,

$$N \leftarrow N + 1 \quad n_N^a = 1 \quad \boldsymbol{\mu}_N^a = \mathbf{A}_k \quad \text{and} \quad \boldsymbol{\sigma}_N^a = \gamma_a \tag{9}$$

where γ_a is a predefined initial standard deviation value. During the prediction process, an unlabeled sample $\mathbf{x}$ is presented to MGA-a, and

```
subroutine_training_GART
{
    Set the training parameters (refer to Part A of Section III, Table I, III and
    IV for the detail of parameters selection).
    Wait for the arriving of the next training pair {Ak, Bk}, or an unlabeled
    vector x for recognition.
    {   if {Ak, Bk} is available for training
        {
          Compute P(Ak | j), P(j | Ak), P(Bk | j),  P(j | Bk) using Eq (1)-(6).
          Select the winner of the competition, J, using (7).
          If no winner found or this is the first training pair presented
          {
              Create a new category using Eq (9).
          }
          Else if a valid winner is found
          {
              Update the weights of J using Eq (8)
          }
        }
        If an unlabeled vector x is available for recognition
        {
            go to subroutine_recognition_GART
        }
    }
}
subroutine_recognition_GART
{
    Compute P(j | x) based on Eq (1)-(6).
    Compute the recognition result of x using Eq (10).
}
```

Fig. 1. Pseudocode for the training and recognition of GART.

TABLE I
SETTING OF THE PARAMETERS FOR ALL EXPERIMENTS

Parameters	Default Value
ρ_b	0 (Regression and Time Series Prediction)
ρ_b	1 (Classification)
γ_b	Same as γ_a
ε_a	0.3
ε_b	0

the prediction of GART is obtained by a distributed posterior probability estimate (similar to the GRNN), as follows:

$$f(\mathbf{x}) = \frac{\sum_{j=1}^{N} \frac{\mu_j^b}{\sigma_j^b} P(j|\mathbf{x})}{\sum_{j=1}^{N} \frac{1}{\sigma_j^b} P(j|\mathbf{x})}. \tag{10}$$

In summary, Fig. 1 shows the pseudocode of the training and recognition phases of GART.

III. EXPERIMENTS AND RESULTS

The performances of GART are evaluated using nine empirical experiments. The experiments in Sections III-A and III-B are based on those in [7] and [11], respectively, while the experiments in Section III-C are based on those in [9]. Unless otherwise stated, the training parameters of MGA-a and MGA-b were set to their "default" values, as in Table I, for all experiments. For each problem, 50 runs were performed using GART (and OSELM), after tuning the two most parameters of GART (γ_a and ρ_a). The statistical significance of the GART results was further evaluated by using the bootstrap technique [20], [21] with 1000 resamplings.

TABLE II
TRAINING PARAMETERS, NUMBER OF CATEGORIES AND BOOTSTRAPPED MEAN, AND 95% CONFIDENCE LIMIT OF GART

	Parameters		No. of Categories	Lower Limit	Upper Limit	**Mean**
	ρa	γa				
Spr	0.6	0.04	1650.5	96.11	96.35	**96.16**
SinE	0.2	0.0005	1908.8	0.0067	0.0069	**0.0068**
Mpg	0.2	0.0075	179.0	0.0706	0.0770	**0.0739**
Aba	0.1	0.04	882.6	0.0793	0.0806	**0.0800**
Cal	0.1	0.02	5860.6	1.1314	0.1319	**0.1316**
Seg	0.1	0.1	634.8	96.61	96.98	**96.80**
Sat	0.01	0.2	1969.4	90.63	90.42	**90.53**
DNA	0.1	1.5	1872.2	88.35	88.35	**88.31**
Mac	0.15	0.01	1521.3	0.0141	0.0143	**0.0142**

Table II shows the values of γ_a and ρ_a, the number of categories, and the results [percentage of accuracy for classification problems; the smallest root mean square error (RMSE) for regression and time series prediction problems] along with their 95% confidence intervals (estimated by the bootstrap technique). The γ_a and ρ_a values were identified based on the minimum error found from several trials. Experiment A showed how parameter selection was conducted.

In Experiments A and B, the model selection and data normalization procedures of OSELM were the same as in [9].[1] Although OSELM could accept a chunk of training samples for online learning, the experiments were focused on the online learning mode where the input samples were received one by one (as explained in Section I).

A. Two Noisy Intertwined Spirals (Spr)

This is a synthetic benchmark problem designed for noisy data classification. The training samples were created based on the procedure

[1]Note that the program code of OSELM was obtained from http://www.ntu.edu.sg/home/egbhuang/.

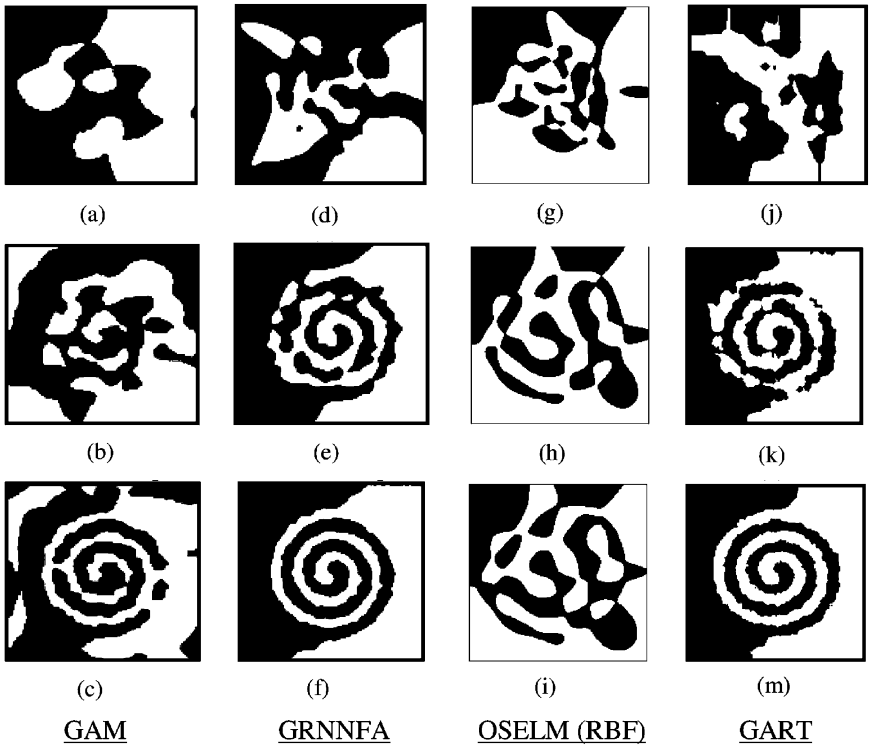

Fig. 2. Evolution of the classification results: Note that (a)–(c) are the results of original GAM (adapted from [11]), (d)–(f) are the results of GRNNFA (adapted from [22]); (g)–(i) are the results of OSELM (RBF); and (j)–(m) are the results of GART. (a) 100 samples; (b) 1000 samples; (c) 10 000 samples; (d) 100 samples; (e) 1000 samples; (f) 10 000 samples; (g) 100 samples; (h) 1000 samples; (i) 10 000 samples; (j) 100 samples; (k) 1000 samples; and (m) 10 000 samples.

TABLE III
PERCENTAGE OF ACCURACY AND NUMBER OF CATEGORIES (IN PARENTHESIS) BASED ON DIFFERENT VALUES OF ρ_a AND γ_a FOR EXPERIMENT A. NOTE THAT $\rho_b = 1, \gamma_b = \gamma_a$, AND $\varepsilon_b = \varepsilon_a = 0$

ρ_a / γ_a	0.3	0.4	0.5	0.6	0.7
0.01	95.17 (2478.3)	95.58 (3343.7)	95.90 (4202.7)	95.93 (5203.8)	95.98 (6365.9)
0.02	94.76 (1277.0)	95.61 (1905.4)	96.07 (2570.2)	96.12 (3337.1)	96.02 (4304.1)
0.03	93.18 (837.1)	95.43 (1335.1)	96.08 (1880.6)	**96.16 (2517.4)**	95.92 (3335.6)
0.04	92.01 (618.1)	*94.70 (1026.5)*	95.40 (1491.9)	96.03 (2047.5)	95.69 (2759.7)

TABLE IV
PERCENTAGE OF ACCURACY AND NUMBER OF CATEGORIES (IN PARENTHESIS) BASED ON DIFFERENT VALUES OF ε_a FOR EXPERIMENT A. NOTE THAT $\rho_a = 0.6$ AND $\gamma_a = 0.03, \rho_b = 1, \gamma_b = \gamma_a$, AND $\varepsilon_b = 0$

ε_a	0.0	0.3	0.6	0.9
Accuracy and No. of categories	*96.16 (2517.4)*	**96.10 (1977.1)**	95.67 (1570.3)	94.99 (1241.5)

in [11]. The samples were generated from two intertwined spirals in 2-D unit square. Each spiral consisted of 97 isotropic Gaussian distributions, with standard deviation of 0.025, centered along the spiral.

Fig. 2 shows the graphical results of 100, 1000, and 10 000 training samples. For comparison purposes, the results of GAM [11] and GRNNFA [22] are included. Both GART and GRNNFA achieved similar results, which are comparatively better than those of GAM and OSELM. GRNNFA used a two-stage training strategy. Fuzzy ART [14] was first used to identify the centers of prototypes. Then, the gradient descent method was used to optimize the width of the kernels (standard deviations). Since the training process (by gradient descent) was repeated for 2000 times [22], it did not fulfill the condition of online learning (as in Section I).

A further experiment was conducted with OSLEM-RBF [9]. After several trials with different random weights, number of hidden neurons, and number of initial training samples, the best result of OSELM-RBF was 84.09%, which was produced using 200 hidden neurons and 4000 initial training samples. However, this is still relatively lower than the best result of GART (96.16%).

The trial-and-error method is commonly used for parameters selection and tuning of many ART-based models e.g., [11], [15], and [23] as well as other learning systems, e.g., [24]. The parameter set that produces the smallest error rate is then used for evaluating the performance using the test set. The same strategy is adopted in this study. To minimize the number of categories created and the category proliferation problem [11], [14], [15], [25], [29], the vigilance parameter ρ_a should not be a large value. As suggested in [11], a large value of γ_a would lead to slower training with fewer categories (but would require the training samples to be presented repeatedly). Hence, ρ_a and γ_a were set to small values to minimize the number of categories created as well as to perform rapid learning. Table III shows the results (averages of 50 runs) for different ρ_a and γ_a settings. As $\rho_a = 0.6$ and $\gamma_a = 0.03$ produced the best result, this setting was selected for further experiments to identify parameter ε_a (note that when ρ_a and γ_a were tuned, ε_a was set to 0).

Parameter ε_a can be considered as an optional user-defined threshold to regulate the network complexity. It is useful especially when the data sets are noisy. Table IV shows the results (averages of 50 runs) for different values of ε_a. When ε_a is larger, the network size is expected to be smaller and classification accuracy to be lower. As can be seen, $\varepsilon_a =$

TABLE V
PERFORMANCE COMPARISON OF EXPERIMENT B (SINE)

Algorithms	Training RMSE		Testing RMSE	
	Mean	STD	Mean	STD
GART	**0.0040**	0.0002	**0.0068**	0.0005
OSELM (RBF)	0.1681	0.0031	0.1695	0.0041
GAPRBF [7]	0.0261	0.0085	0.0269	0.0087
MRAN [7]	0.0477	0.0205	0.0496	0.0212
RANEKF [7]	0.0265	0.0117	0.0275	0.0124
RAN [7]	0.0659	0.0107	0.0673	0.0107
SVR [7]	0.2591	0.0033	0.2629	0.0056

TABLE VI
SPECIFICATION OF BENCHMARKS DATA SETS FOR EXPERIMENTS IN [8]

Datasets	# Attributes	# Class or output	# Training Samples	# Testing Samples
Mpg	7	1	320	72
Aba	9	1	3,000	1,1177
Cal	8	1	8,000	12,640
Seg	18	7	1,500	810
Sat	36	6	4,435	2,000
DNA	180	3	2,000	1,186
Mac	4	1	4,000	500

0.3 yielded a smaller network structure (21.46% fewer categories), but with only 0.06% drop in accuracy, as compared with those of $\varepsilon_a = 0$. Note that the ε-insensitive loss function is equivalent to the Laplacian loss function when $\varepsilon_a = 0$ [18]. With $\varepsilon_a = 0.9$, the classification accuracy rate dropped by 1.17%, but with 50.57% reduction in network size. These results indicate that there is a tradeoff between network accuracy (performance) and network size (compression). This tradeoff is observed in other ART-based networks [15], [22], [23], [27], [28] as well as other neural network models [4], [26]. Note that for simplicity, ε_a was set to 0.3 for all other experiments.

B. SinE

In this experiment, GART was used to approximate a rapidly changing continuous function known as "SinE" [7], i.e.,

$$y = 0.8\exp(-0.2x)\sin(10x). \tag{11}$$

A total of 50 runs were conducted with 3000 training samples in each run. The samples comprised randomly generated x values (between 0 and 10), with the respective y values. Another 1500 test samples were generated using the same procedure. The results of GART and OSELM were compared with those of GAPRBF, MRAN, RANEKF, RAN, and support vector regression (SVR), as summarized in Table V. The OSELM result was obtained using 200 RBF neurons and 400 samples for initial training. As shown in Table V, GART produced smaller RMSE values for both training and test sets as compared with those of other models.

C. Experiments in [9]

Seven benchmark experiments, as presented in [9], i.e., Auto-MPG (Mpg), Abalone (Aba), California Housing (Cal) (regression), Image Segmentation (Seg), Satellite Image (Sat), DNA (classification), and Mackey Glass (Mac) (time series prediction), were evaluated. Table VI summarizes the specifications of these problems. Further details of the problems and the data sets used were explained in [9].

The results of regression problems are given in Table VII. From the three problems, GART achieved the best test results in Auto-MPG and California Housing, and the best training results in all problems. The classification results are given in Table VIII. Out of the three classification problems, GART achieved the best results in both training and test

TABLE VII
COMPARISON AMONG GART, OSELM, AND OTHER SEQUENTIAL LEARNING ALGORITHMS FOR REGRESSION PROBLEMS

Datasets	Algorithms	Average RMSE	
		Training	Test
Mpg	*GART*	**0.0417**	**0.0739**
	OSELM (Sigmoid) [9]	0.0680	0.0745
	OSELM (RBF) [9]	0.0696	0.0759
	Stochastic BP [9]	0.1112	0.1028
	GAPRBF [7]	0.1144	0.1404
	MRAN [7]	0.1086	0.1376
	RANEKF [7]	0.1088	0.1387
	RAN [7]	0.2923	0.3080
Aba	*GART*	**0.0646**	0.0800
	OSELM (Sigmoid) [9]	0.0754	**0.0777**
	OSELM (RBF) [9]	0.0759	0.0783
	Stochastic BP [9]	0.0996	0.0972
	GAPRBF [7]	0.0963	0.0966
	MRAN [7]	0.0836	0.0837
	RANEKF [7]	0.0738	0.0794
	RAN [7]	0.0931	0.0978
Cal	*GART*	**0.0683**	**0.1316**
	OSELM (Sigmoid) [9]	0.1303	0.1332
	OSELM (RBF) [9]	0.1321	0.1341
	Stochastic BP [9]	0.1688	0.1704
	GGAPRBF [8]	0.1417	0.1386
	MRAN [8]	0.1598	0.1586
	RANEKF [8]	0.0736	0.1495
	RAN [8]	0.1083	0.1531

TABLE VIII
COMPARISON BETWEEN GART, OSELM, AND OTHER SEQUENTIAL LEARNING ALGORITHMS FOR CLASSIFICATION PROBLEMS

Datasets	Algorithms	Average Accuracy	
		Training	Testing
Seg	*GART*	**99.49**	**96.80**
	OSELM (sigmoid) [9]	97.00	94.88
	OSELM (RBF) [9]	96.65	94.53
	Stochastic BP [9]	83.71	82.55
	GAPRBF [9]	-	89.93
	MRAN [9]	-	93.30
Sat	*GART*	**98.04**	**90.53**
	OSELM (sigmoid) [9]	91.88	88.33
	OSELM (RBF) [9]	93.18	89.01
	Stochastic BP [9]	85.23	83.75
	MRAN [9]	-	86.36
DNA	GART	**100**	88.31
	OSELM (sigmoid) [9]	95.79	93.43
	OSELM (RBF) [9]	96.12	**94.37**
	Stochastic BP [9]	85.64	82.11
	MRAN [9]	-	86.85

TABLE IX
COMPARISON BETWEEN GART, OSELM, AND OTHER SEQUENTIAL LEARNING ALGORITHMS FOR MACKEY GLASS TIME SERIES PREDICTION PROBLEM

Algorithms	Average RMSE	
	Training	Testing
GART	**0.0061**	**0.0142**
OSELM (sigmoid) [9]	0.0177	0.0183
OSELM (RBF) [9]	0.0184	0.0186
GGAPRBF [8]	0.0700	0.0368
MRAN [8]	0.1101	0.0337
RANEKF [8]	0.0726	0.0240
RAN [8]	0.1006	0.0466

sessions for Image Segmentation and Satellite Image. However, GART did not perform well for the DNA problem. It achieved 100% accuracy during training, with 1872.2 (average of 50 runs) number of categories. This implied that GART might be overtrained. This was likely because the data set has a large number of input attributes, and all of them are binary values. To further examine the performance of GART, another DNA experiment, based on 60 attributes as reported in [30] and [31], was conducted. GART achieved average (of 50 runs) classification rates of 99.22% and 91.29% for the training and test sets, respectively. The number of categories created was 904.1 (average of 50 runs).

The results of the Mackey Glass time series prediction problem are given in Table IX. Comparing with OSELM and other sequential RBF networks, GART achieved smaller RMSE values in both training and test sessions.

IV. Summary

An ART-based online learning neural network called GART has been introduced. The efficiency of GART has been evaluated and compared with those from a variety of online learning models using nine benchmark experiments. The results show that GART is able to achieve better performances in seven out of the nine experiments. For future work, dynamic initialization of the standard deviation of the MGA modules is to be investigated to improve the robustness of GART.

References

[1] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Clarendon, 1995.

[2] D. E. Rumellhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Parallel Distribution Process: Explanations Microstructure of Cognition*, vol. 1, pp. 318–362, 1986.

[3] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–309, Mar. 1991.

[4] J. Platt, "A resource-allocating network for function interpolation," *Neural Comput.*, vol. 3, pp. 213–225, 1991.

[5] V. Kadirkamanathan and M. Niranjan, "A function estimation approach to sequential learning with neural networks," *Neural Comput.*, vol. 5, pp. 954–975, 1993.

[6] L. Yingwei, N. Sundararajan, and P. Saratchandran, "Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm," *IEEE Trans. Neural Netw.*, vol. 9, no. 2, pp. 308–318, Mar. 1998.

[7] G.-B. Huang, P. Saratchandran, and N. Sundararajan, "An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 34, no. 6, pp. 2284–2292, Nov. 2004.

[8] G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation," *IEEE Trans. Neural Netw.*, vol. 16, no. 1, pp. 57–67, Jan. 2005.

[9] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithms for feedforward network," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1411–1423, Nov. 2006.

[10] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Real-time learning capability of neural networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 863–878, Jul. 2006.

[11] J. R. Williamson, "Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps," *Neural Netw.*, vol. 9, no. 5, pp. 881–897, 1996.

[12] D. F. Specht, "A general regression neural network," *IEEE Trans. Neural Netw.*, vol. 2, no. 6, pp. 568–576, Nov. 1991.

[13] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, pp. 1065–1076, 1962.

[14] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Netw.*, vol. 4, pp. 759–771, 1991.

[15] G. A. Carpenter, S. Grossberg, N. Markuzon, and J. H. Reynolds, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 698–713, Sep. 1992.

[16] K. S. Yap, C. P. Lim, I. Zainal Abidin, and M. Malik, "A hybrid neural network for time series prediction," in *Proc. 3rd Int. Conf. Artif. Intell. Eng. Technol.*, 2006, vol. 1, pp. 451–456.

[17] K. S. Yap, I. Zainal Abidin, and C. P. Lim, "Short term load forecasting using a hybrid neural network," in *Proc. IEEE 1st Int. Conf. Power Energy*, 2006, vol. 1, pp. 123–128.

[18] W. Chu, S. S. Keerthi, and C. J. Ong, "Bayesian support vector regression using a unified loss function," *IEEE Trans. Neural Netw.*, vol. 15, no. 1, pp. 29–44, Jan. 2004.

[19] J. Xu and X.-G. Zhang, "Kernels based on weighted Levenshtein distance," in *Proc. Int. Joint Conf. Neural Netw.*, Budapest, Hungary, Jul. 25–29, 2004, vol. 4, pp. 3015–3018.

[20] B. Efron, "Bootstrap methods: Another look at the jackknife," *Ann. Statist.*, vol. 7, pp. 1–26, 1979.

[21] B. Efron, "Nonparametric standard errors and confidence intervals," *Can. J. Statist.*, vol. 9, pp. 139–172, 1981.

[22] E. W. M. Lee, C. P. Lim, R. K. K. Yuen, and S. M. Lo, "A hybrid neural network model for noisy data regression," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 34, no. 2, pp. 951–960, Apr. 2004.

[23] R. Andonie and L. Sasu, "Fuzzy ARTMAP with input relevance," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 929–941, Jul. 2006.

[24] J. Leng, L. Jain, and C. Fyfe, "Simulation and reinforcement learning with soccer agents," *J. Multi-Agent Grid Syst.*, vol. 4, no. 4, Dec. 2008, to be published.

[25] B. Moore, "ART 1 and pattern clustering," in *Proc. Connectionist Models Summer School*, San Mateo, CA, 1988, pp. 174–185.

[26] L. Mah and K. Khorasi, "Constructive feedforward neural networks using hermite polynomial activation functions," *IEEE Trans. Neural Netw.*, vol. 16, no. 4, pp. 821–833, Jul. 2005.

[27] N. Srinivasa, "Learning and generalization of noisy mappings using a modified PROBART neural network," *IEEE Trans. Signal Process.*, vol. 45, no. 10, pp. 2533–2550, Oct. 1997.

[28] S. Marriott and R. S. Harrison, "A modified fuzzy ARTMAP architecture for the approximation of noisy mappings," *Neural Netw.*, vol. 8, no. 4, pp. 619–641, 1995.

[29] E. Gomez-Sanchez, Y. A. Dimitriadis, J. M. Cano-Izquierdo, and J. Lopez-Coronado, "μ ARTMAP: Use of mutual information for category reduction in fuzzy ARTMAP," *IEEE Trans. Neural Netw.*, vol. 13, no. 1, pp. 58–69, Jan. 2002.

[30] M. Ankerst, M. Ester, and H. P. Kriegel, "Towards an effective cooperation of the user and the computer for classification," in *Proc. 6th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, 2000, pp. 179–189.

[31] The Laboratory of Artificial Intelligence and Decision Support, University of Porto, "DNA dataset description," Porto, Portugal, Jan. 2006 [Online]. Available: http://www.liaad.up.pt/old/statlog/datasets/dna/dna.descri.html