2009 Ninth International Conference on Intelligent Systems Design and Applications

Classification by Evolutionary Generalized Radial Basis Functions

A. Castaño[1], C. Hervás-Martínez[2], P. A. Gutierrez[2], F. Fernández-Navarro[2], M. M. García[3]

[1]Department of Informatics, University of Pinar del Rio, Pinar del Rio, Cuba.
adiel@info.upr.edu.cu
[2]Department of Computer Science of the University of Córdoba, Campus de Rabanales, 14071, Córdoba, Spain.
{chervas, pagutierrez,i22fenaf}@uco.es
[3]Department of Computer Science, University of Las Villas, Santa Clara, Cuba.
mmgarcia@uclv.edu.cu

Abstract

This paper proposes a novelty neural network model by using generalized kernel functions for the hidden layer of a feed forward network (Generalized Radial Basis Functions, GRBF), where the architecture, weights and node typology are learned through an evolutionary programming algorithm. This new kind of model is compared with the corresponding models with standard hidden nodes: Product Unit Neural Networks (PUNN), Multilayer Perceptrons (MLP) and the RBF neural networks. The methodology proposed is tested using six benchmark classification datasets from well-known machine learning problems. Generalized basis functions are found to present a better performance than the other standard basis functions for the task of classification.

1. Introduction

The simplest method for classification provides the class label given its observation via linear functions in predictor variables. This process of model fitting is quite stable, resulting in low variance but a potentially high bias. Frequently, in a real-problem of classification, we cannot make the stringent assumption of additive and purely linear effects of the variables. A traditional technique to overcome these difficulties is augmenting/replacing input vector with new variables, the basis functions, which are transformations of the input variables, and then a linear model is used in this new space of derived input features. Once the number and structure of the basis functions have been determined, the models are linear in these new variables and the fitting is a standard procedure.

Different types of neural networks, NNs, are nowadays being used for classification purposes [1], including, among others: multilayer perceptron neural networks (MLP) where the transfer functions are Sigmoidal Unit (SU) basis functions; Radial Basis Function (RBF) neural networks with kernel functions where the transfer functions are usually Gaussian [2]; General Regression Neural Networks (GRNN) proposed by Specht [3]; and a class of multiplicative NNs, namely Product Unit Neural Networks (PUNNs), [4,5]. A characteristic that distinguishes all these models is the combination of transfer and activation functions used in the hidden layer of the neural network. In the rest of the paper, this pair of functions is referred to as the basis function.

The RBF network can be considered a local average procedure and the improvement in its approximation ability as well as in the construction of its architecture has drawn a lot of attention. Bishop [2] concluded that an RBF network can provide a fast linear algorithm capable to represent complex non-linear mappings. An RBF classifier is a three-layer neural network model, in which an K-dimensional input vector $\mathbf{x} = (x_1, x_2,\ldots, x_K)$ is broadcast to each of M neurons in the hidden layer. The most common RBF is represented by a Gaussian function, but when dimensionality grows and/or when data is concentrated in boundaries of the K dimensional space, standard Gaussian basis function lacks its performance. One of the most evident reasons is that when dimensionality grows, the distances' average from a RBF to the instances that are covered by its corresponding centre grows.

With the aim of alleviating this problem associated to the high dimensionality of the input space, this work

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.29

IEEE computer society

evaluates the accuracy obtained by a special class of RBF NNs, namely generalized radial basis neural networks. The training of these networks is performed by a specific evolutionary algorithm, where the principal issue is the stablishment of a method to choose adequate values for the principal parameters of the generalized basis functions.

Section 2 introduces the generalized RBF Neural Networks. Section 3 formally presents the new generalized radial basis function neural network model for classification considered in this work. In section 4, the main characteristics of the algorithm used for training the model are described. Section 5 presents the experiments carried out and discusses the results obtained. Finally, Section 6 completes the paper with the main conclusions and future directions suggested by this study.

2. Neural Networks based on Generalized Basis Functions

RBF neural networks have been used in the most varied domains, from function approximation, to pattern classification, time series prediction, data mining, signals processing, and nonlinear system modelling and control. They have some useful properties which render them suitable for modelling and control. First, such networks are universal approximators [6]. In addition, they belong to a class of linearly parameterized networks where the network output is connected to tuneable weights in a linear manner. Due to their functional approximation capabilities, RBF networks have been seen as a good solution for interpolation problems. They are also able to provide regularized solutions for ill-posed problems [7]. RBF can be considered a local average procedure, and the improvement in both its approximation ability as well as in the construction of its architecture has been note-worthy [8]. One of the most important issues is network learning, i.e., the optimization of adjustable parameters, which include centre vectors, radii (or widths of the Gaussian distributions), and linear output weights connecting the RBF hidden nodes to the output nodes. Another important issue is the determination of the network's structure or the number of RBF hidden nodes based on the parsimonious principle [8,9]. So, the number and positions of basis functions, which correspond to the neurons in the hidden layer of the network, have an important influence on the performance of the RBF neural net. Both problems have been tackled using a variety of approaches. For instance, the number and position of the RBFs may be fixed and defined a priori [9]; they may be determined by unsupervised clustering algorithms [10]; or through a supervised learning scheme that includes growing and pruning procedures [11]; or they can be evolved using evolutionary algorithms [12,13], or hybrid algorithms [14]. One of the most common RBF model is represented by a Gaussian function where the output depends on the distance between the instance and the centre of the RBF. This distance can be formulated in different ways; the most common formulation is the Euclidean distance, but when dimensionality grows and/or when data is concentrated in boundaries of the K dimensional space, standard Gaussian basis functions lack their performance. To prevent the effects observed for standard Gaussian RBFs, these basis functions can be generalized by means of replacing the usual exponent 2 by a new parameter τ which can relax or contract the Gaussian. In this way, Generalized Gaussian kernels basis functions, GRBF, are defined using the following expression [15]:

$$B_j(\mathbf{x}, \mathbf{w}_j) = \exp\left(-\frac{\left\|\mathbf{x} - \mathbf{c}_j\right\|^{\tau_j}}{r_j^{\tau_j}} \right) \quad (1)$$

where $\mathbf{x}_i = (x_{1i}, ..., x_{Ki})$ is the random vector of measurements, $\mathbf{w_j} = (w_{j0}, w_{j1}, ..., w_{jK}, w_{j(K+1)})$, $\mathbf{c_j} = (w_{j1}, ..., w_{jK})$, $r_j = w_{j0}$ and $\tau_j = w_{j(K+1)}$ are, respectively, the centre, the width and the exponent of the j-th generalized radial basis function and K is the number of inputs. These basis functions allow a better matching between the shape of the kernel and the distribution of the distances, since the τ parameter provokes concavity and or convexity around the point where radium = r (see Figure 1). Indeed standard Gaussian lacks its performance when the mean of distances distribution separates from zero but Generalized Gaussian is able to adapt to this difficulty.

3. Generalized Radial Basis Functions for Classification

In a classification problem, measurements x_i, $i = 1, 2, ..., K$, of a single individual (or object) are taken, and the individuals are to be classified into one of the J classes based on these measurements. A training sample $D = \{(\mathbf{x}_n, \mathbf{y}_n); n = 1, 2, ..., N\}$ is available, where $\mathbf{x}_n = (x_{1n}, ..., x_{kn})$ is the random vector of measurements taking values in $\Omega \subset R^K$, and $\mathbf{y}_n$ is the class level of the n-th individual, where the common technique of representing class levels using a "1-of-J" encoding vector is adopted, $\mathbf{y} = \left(y^{(1)}, y^{(2)} ..., y^{(J)}\right)$, and the Correctly Classified Rate or accuracy of the classifier is defined by $CCR = \frac{1}{N}\sum_{n=1}^{N} I(C(\mathbf{x}_n) = \mathbf{y}_n)$, where $I(.)$ is the zero-one lost function. A good classifier tries to achieve the highest possible CCR in a given problem.

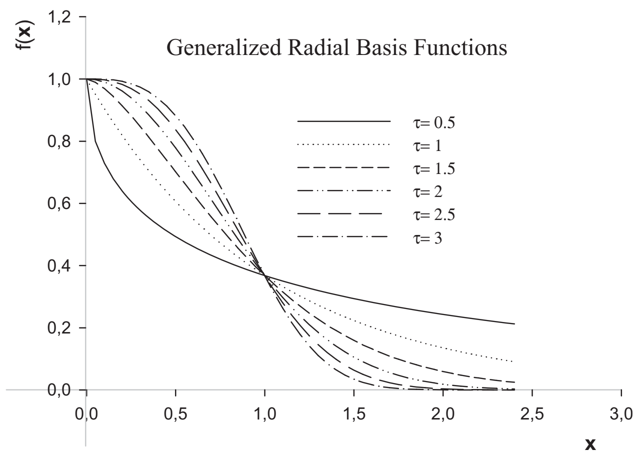

Fig. 1. Generalized Gaussian with r=1, and different τ values

The softmax activation function [4] is considered as that given by:

$$g_l\left(\mathbf{x}, \boldsymbol{\theta}_l\right) = \frac{\exp f_l\left(\mathbf{x}, \boldsymbol{\theta}_l\right)}{\sum_{l=1}^{J} \exp f_l\left(\mathbf{x}, \boldsymbol{\theta}_l\right)}, \; l = 1, 2, ..., J \qquad (2)$$

where J is the number of classes in the problem, $f_l\left(\mathbf{x}, \boldsymbol{\theta}_l\right)$ the output of the j output neuron for pattern $\mathbf{x}$ and $g_l\left(\mathbf{x}, \boldsymbol{\theta}_l\right)$ is the probability a pattern $\mathbf{x}$ has of belonging to class j. The model to estimate the function $f_l\left(\mathbf{x}, \boldsymbol{\theta}_l\right)$ is given by a kernel basis function model:

$$f_l\left(\mathbf{x}, \boldsymbol{\theta}_l\right) = \sum_{j=1}^{m} \beta_j^l \exp\left(-\frac{\left\|\mathbf{x} - \mathbf{c}_j\right\|^{\tau_j}}{r_j^{\tau_j}}\right) \qquad (3)$$

where, if we use a generalized basis function, the activation function of the j-th radial basis function, $B_j(\mathbf{x}, \mathbf{w})$ can be defined as equation (1).

Taking into account that the outputs of the neurons are interpreted from the point of view of probability through the use of the softmax activation function in equation (2), it can be seen that the class predicted by the neural net corresponds to the neuron in the output layer whose output value is the greatest. The optimum classification rule $C(\mathbf{x})$ is the following:

$$C(x) = \hat{l}, \text{ where } \hat{l} = \operatorname{argmax}_l g_l(x, \hat{\theta}), \text{ for } l = 1, 2, ..., J \quad (4)$$

The function used to evaluate a classification model is the function of cross-entropy error and it is given by the following expression for J classes:

$$l(\boldsymbol{\theta}) = \sum_{n=1}^{N}\left[-\sum_{i=1}^{J} y_n^{(l)} f_l(\mathbf{x}_n, \boldsymbol{\theta}_l) + \log \sum_{l=1}^{J} \exp f_l(\mathbf{x}_n, \boldsymbol{\theta}_l)\right] \qquad (5)$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_J)$. As can be observed in the next section, the proposed algorithm returns to the best cross-entropy individuals as feasible solutions. Moreover, because of the normalization condition:

$$\sum_{l=1}^{J} g_l\left(\mathbf{x}, \boldsymbol{\theta}_l\right) = 1 \qquad (6)$$

and the probability for one of the classes does not need to be estimated.

The error surface associated with the model is very convoluted with numerous local optima and the Hessian matrix of the error function $l(\boldsymbol{\theta})$ is, in general, indefinite. Moreover, the optimal number of basis functions in the model (i.e. the number of hidden nodes in the neural network) is unknown, and, in this case, the GRBF are not Mercer kernels, i.e., they are not positive semi-definite for all values of the τ parameter [16]. In this way, the optimisation problem will generally not be convex. Thus, we determine the estimation of the vector parameters $\hat{\boldsymbol{\theta}}$ by means of an evolutionary algorithm.

4. Evolutionary Algorithm

The evolutionary algorithm, EA, designs the structure and learns the weights of the GRBF neural networks. The search begins with an initial population of GBFR neural networks, and, in each iteration, the population is updated using a population-update algorithm. The population is subject to the operations of replication and mutation. The general structure of the EA is similar to the structure of the one presented in [16], but with several significant modifications. In the current approach, $l(\boldsymbol{\theta})$ is the error function of an individual g of the population, where g is a GRBF neural network, which is given by the multivaluated function $\mathbf{g}\left(\mathbf{x}, \boldsymbol{\theta}\right) = \left(g_1\left(\mathbf{x}, \boldsymbol{\theta}_1\right), ..., g_l\left(\mathbf{x}, \boldsymbol{\theta}_J\right)\right)$. The fitness measure is a strictly decreasing transformation of the entropy error $l(\boldsymbol{\theta})$, given by $A(g) = \frac{1}{1 + l(\boldsymbol{\theta})}$. The severity of mutations depends on the temperature $T(g)$ of the GRBF neural network model, which is defined by $T(g)=1-A(g), \quad 0\le T(g)\le 1$.

For GRBF hidden nodes, the connections between the input layer and hidden layer are initialized using a clustering algorithm, so the EA can start the evolutionary process with well positioned centers. The main idea is to cluster input data in M groups, M being the number of hidden GRBF neurons. Therefore, each hidden GRBF neuron can be positioned in the centroid of its corresponding cluster. A first algorithm modification consists on GRBF's radii initialization, where the determination of the initial τ and r values are intimately related to the distribution of the distances and can be set according to the specificities of that distribution. The method to choose adequate values for τ and r is based on: largest or "farest" distances (d_{F}, the 5-th percentile of the distribution) must be mapped to lower values of the probability. Then the d_{F} values can be approximately calculated as $\mu + 1.645\sigma$, where μ is the mean of individual´s distance to the centroid and σ is the standard deviation of these distances. Then, τ and r can be calculated as follows by solving two different equations:

$$\exp\left(-\left(\frac{d_{\mathrm{F}}}{r}\right)^{\tau}\right)=0.05,$$

$$\exp\left(-\left(\frac{\mu}{r}\right)^{\tau}\right)=0.5.$$

The solution of these equations is:

$$\tau=\frac{\ln\left(\dfrac{\ln(0.05)}{\ln(0.5)}\right)}{\ln\left(\dfrac{d_{\mathrm{F}}}{\mu}\right)} \text{ and } r=d_{\mathrm{F}}(-\ln(0.05))^{-\frac{1}{\tau}}$$

The most critical part of these equations is determined by the μ value and the "farest" distance (d_{F}). For that, we use estimators of the μ and d_{F} associated to the statistic distribution of the distances between the centroids and the individuals in the cluster.

In every generation, a parametric mutation is accomplished for each coefficient w_{ji} or β_j^l of the model with Gaussian noise, where the variances of the normal distribution are updated throughout the evolution of the algorithm. Once the mutation is performed, the fitness of the individual is recalculated and a usual simulated annealing process is applied. First, the link weights are mutated by adding a value $\xi\in N(0,\alpha\cdot T(g))$. r is mutated in the same way, adding a value $\eta\in N(0,\alpha\cdot T(g))$. The variance α is updated throughout the evolution of the algorithm. There are different methods to update the variance. We use the 1/5 success rule of Rechenberg, one of the simplest methods [17]. The modification of GRBFs is very sensible to τ variation. Indeed, when τ is near to the interval [0, 2.5], a τ variation changes drastically the contraction of GRBF basis function. On the other hand, when τ >> 2.5, the same τ variation does not change drastically the generalized Gaussian. Due to this behavior, τ modification value must depend on the desired effect (see Fig. 2).

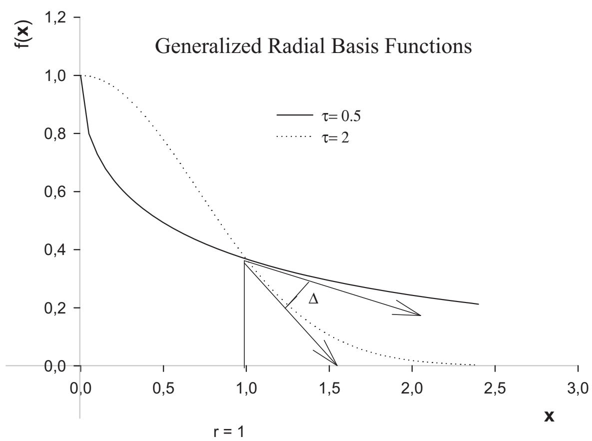

Fig. 2. Generalized Gaussian with r= 1, and different τ values

To define this desired effect, the τ mutation is formulated as:

$$\tau_{n+1}=\frac{e\cdot r(\tau_n-e\cdot r\tan(\Delta))}{e\cdot r+\tau_n\tan(\Delta)}$$

where e is the Euler's constant and Δ is the angle's variation that must be produced on the tangent of the curve associated to the generalized function at the point where the radium is r (see fig. 2).

On the other hand, structural mutation implies a modification in the neural network structure and allows explorations in different regions in the search space while helping to keep up the diversity of the population. There are two structural mutations: node deletion and node addition. These mutations are applied sequentially to each network [16].

In order to define the topology of the neural networks generated in the evolution process, we consider three parameters: m, M_{E} and M_{I}. They respectively correspond to the minimum and the maximum number of hidden nodes in the whole evolutionary process and the maximum number of hidden nodes in the initialization process. In order to obtain an initial population formed by models simpler than the most complex model possible, parameters must fulfil the condition $m \le M_{\mathrm{I}} \le M_{\mathrm{E}}$.

We generate $10N_{\mathrm{P}}$ networks, where $N_{\mathrm{P}} = 1,000$ is the number of population networks during the evolutionary process. Then we select the best N_{P} neural networks. To generate a network, the number of nodes in the hidden layer is taken from a uniform distribution in the interval $[m, M_{\mathrm{I}}]$. For hidden nodes, the number of connections is always $K+2$, where K is the number of inputs, since these connections represent, respectively, the centre, the width and the exponent of each generalized radial basis function. The number of connections between each hidden node and the output layer is determined from a uniform distribution in the interval $(0, J-1]$. The stop criterion is reached if one of the following conditions is fulfilled: a maximum number of generations is reached or the variance of the fitness of the best ten percent of the population is less than 10^{-4}. The number of nodes that can be added or removed in a structural mutation is within the $[1,2]$ interval.

5. Experiments

In order to analyze the performance of the Generalized Radial Basis neural networks, six datasets in the UCI repository have been tested. The experimental design was conducted using a holdout cross-validation procedure with $3n/4$ instances for the training dataset and $n/4$ instances for the generalization dataset, where n is the size of the dataset. All parameters of the NNEP algorithm are common for all problems, except the m, M_{I}, M_{E} values and the number of generations, which are represented in Table 1 together the main characteristics of each dataset.

For each dataset, we will perform an analysis of the results obtained using GRBF basis functions and other basis functions commonly used in neural network models for classification. Table 2 shows the mean value and standard deviation for the training and generalization sets, of the Correctly Classified Rate (CCR) of the nets obtained in 30 runs of the experiment. It can be seen in Table 2 that the GRBF basis function models present the best results for CCR_G. It is interesting to note that the higher differences favouring GRBF models with respect to the other models are obtained for those datasets with a high number of characteristics (Card, German, Ionosphere and Zoo).

6. Conclusions

The models proposed, formed by Generalized Radial Basis Functions as transfer functions, are a viable alternative for obtaining more accurate classifications. These models have been designed with an evolutionary algorithm constructed specifically for taking into account the characteristics of this kernel model. The evaluation of the model and the algorithm for the six datasets considered, showed results that are comparable to those of other basis function neural networks models [18]. GRBF models obtain higher accuracy when they are compared to the rest of basis functions for those datasets with a high number of characteristics.

Acknowledgements

This work has been partially subsidized by TIN 2008-06681-C06-03 project of the Spanish Ministerial Commission of Science and Technology (MICYT), FEDER funds and the P08-TIC-3745 project of the "Junta de Andalucía" (Spain).

References

[1] R.P. Lippmann, Pattern classification using neural networks, IEEE Communications Magazine 27 (1989) 47-64.
[2] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press 1995.
[3] D.F. Specht, A General Regression Neural Network, IEEE Transactions on Neural Networks 2(6) 568-576. 1991.
[4] R. Durbin, and D. Rumelhart, Products Units: A computationally powerful and biologically plausible extension to backpropagation networks, Neural Computation 1, 133-142. 1989.
[5] C. Hervás, F.J. Martinez, and P.A. Gutiérrez, Classification by means of Evolutionary Product-Unit Neural Networks, in: Proceedings of the International Joint Conference on Neural Networks, Canada, 1525-1532. 2006.
[6] J. Park, and I.W. Sandberg, Universal Approximation Using Radial Basis Function Networks, Neural Computation 3 (2) 246-257. 1991.

[7] M. J. L. Orr, Regularisation in the Selection of Radial Basis Function Centres, Neural Computation 7(3) 606-623. 1995.
[8] C. M. Bishop, Improving the generalization properties of radial basis function neural networks. Neural Computation 3 (4) 579-581. 1991.
[9] K. J. Hunt, D. Sbarbaro, R. Zbikowski, and P.J. Gawthrop, Neural networks for control system - a survey, Automatica, 28, 1083-1112. 1992.
[10] C. Darken, and J. Moody, Fast adaptive K-means clustering: some empirical results, in: Proceedings of the IEEE INNS International Joint Conference On Neural Networks, 233-238. 1990
[11] G. B. Huang, P. Saratchandran, and N. Sundararajan, A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation, IEEE Transactions on Neural Networks 16, (1) 57-67. 2005.
[12] F. H. F. Leung, H. K. Lam, S. H. Ling, and P. K.S. Tam, Tuning of the structure and parameters of a neural network using an improved genetic algorithm, IEEE Transactions on Neural Networks 14 (1) 79-88. 2003.
[13] L .N. Castro, E.R. Hruschka, and R.J.G.B. Campello, An Evolutionary Clustering Technique with Local Search to Design RBF Neural Network Classifiers, in: Proceedings of the IEEE International Joint Conference on Neural Networks, 2083-2088. 2004
[14] Z. Q. Zhao, and D.S. Huang, A mended hybrid learning algorithm for radial basis function neural networks to improve generalization capability, Applied Mathematical Modelling 31, 1271-1281. 2007
[15] D. Francois. High-dimensional Data Analysis. VDM Verlag. 2008
[16] F. J. Martínez-Estudillo, C. Hervás-Martínez, P. A. Gutiérrez-Peña, A. C. Martínez-Estudillo. Evolutionary Product-Unit Neural Networks for Classification. Neurocomputing. 72, 549-562. (2008).
[17] Rechenberg. Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Frommann-Holzboog, Verlag, Stuttgart, 1973.
[17] V. Vapnik, and N. Vladimir. Statistical learning theory. Wiley Interscience. 1998
[18] P. A. Gutiérrez, C. Hervás, M. Carbonero and J. C. Fernández. Combined Projection and Kernel Basis Functions for Classification in Evolutionary Neural Networks. Neurocomputing. 2009.

Table 1. Main characteristics of each dataset tested and non-common parameter values.

Dataset	# Ins.	R	B	N	# Inp.	Distribution	# Clas.	m	M_I	M_E	# gen.
Card	690	6	4	5	51	(307, 383)	2	1	2	3	50
German	1000	6	3	11	61	(700,300)	2	2	3	4	300
Glass	214	9	-	-	9	(17,76,13,29,70,9)	6	7	8	9	500
Ionos.	351	33	1	-	34	(126,225)	2	3	4	5	300
Newthy	215	5	-	-	5	(150,35,30)	3	1	1	4	100
Zoo	101	1	15	-	16	(41,20,5,13,4,8,10)	7	2	3	3	400

R: Real; B: Binary; N: Nominal; #Ins.: number of instances; #Inp.: number of inputs; #Clas.: number of classes; #gen.: number of generations.

Table 2. Statistical results in training and generalization CCR for the six datasets considered and 30 executions of the EP algorithm using different basis functions. The best result in the generalization set has been represented in bold face.

		Training	Generalization			Training	Generalization
Dataset	Func.	Mean± SD	Mean± SD	Dataset	Func.	Mean± SD	Mean± SD
Card	GRBF	89.10±0.85	**87.94±1.38**	German	GRBF	76.57±1.35	**74.33±2.77**
	RBF	78.51±1.90	76.69±3.33		RBF	73.71±0.88	71.69±1.32
	PU	84.40±2.02	87.50±2.75		PU	74.13±1.37	71.24±1.52
	SU	86.82±0.94	87.71±1.42		SU	81.21±1.39	73.07±1.64
Glass	GRBF	72.88±2.96	**68.93± 5.19**	Ionos.	GRBF	99.11± 0.47	**93.75±1.76**
	PU	75.90±4.74	65.16±4.17		PU	96.79±1.13	91.15±2.20
	SU	75.22±2.52	67.67±3.49		SU	98.83±0.75	92.61±1.56
	RBF	66.29±2.81	64.91±4.74		RBF	91.39±1.27	90.42±2.60
Newth.	GRBF	99.94±0.19	**97.10±1.93**	Zoo	GRBF	100±0.00	**94.93± 2.77**
	RBF	95.67±0.62	95.00±2.01		RBF	78.46±2.73	75.07±5.00
	PU	99.25±0.55	96.85±2.71		PU	98.20±1.74	94.80±4.48
	SU	98.72±0.65	94.88±2.26		SU	99.34±1.02	92.67±4.34