Evolutionary q-Gaussian Radial Basis Functions for Binary-Classification

F. Fernández-Navarro[1], C. Hervás-Martínez[1], P.A. Gutiérrez[1], M. Cruz-Ramírez[1], and M. Carbonero-Ruz[2]

[1] Department of Computer Science and Numerical Analysis, University of Cordoba, Rabanales Campus, Albert Einstein building 3º floor, 14071, Córdoba, Spain
[2] Department of Management and Quantitative Methods, ETEA, Escritor Castilla Aguayo 4, 14005, Cordoba, Spain

Abstract. This paper proposes a Radial Basis Function Neural Network (RBFNN) which reproduces different Radial Basis Functions (RBFs) by means a real parameter q, named q-Gaussian RBFNN. The architecture, weights and node topology are learnt through a Hybrid Algorithm (HA) with the *iRprop+* algorithm as the local improvement procedure. In order to test its overall performance, an experimental study with eleven datasets, taken from the UCI repository is presented. The RBFNN with the q-Gaussian is compared to RBFNN with Gaussian, Cauchy and Inverse Multiquadratic RBFs.

1 Introduction

Different types of neural networks, are being used for classification purposes [1], including, among others: Multilayer Perceptron Neural Networks (MLPNN) where the transfer functions are Sigmoidal Unit Basis Functions; Radial Basis Function Neural Networks (RBFNNs) with kernel functions where the transfer functions are usually Gaussian [2]; Product Unit Neural Networks (PUNNs)[3] with multiplicative units, or Neural Network where the hidden layer is composed by a mixture of basis functions [4].

We focus on RBFNNs which have been succesfully employed in different pattern recognition problems in the last years [5]. There are several common types of functions used as the transfer functions, for example, the standard Gaussian (SRBF), $\phi(z) = e^{-z}$, the Multiquadratic (MRBF), $\phi(z) = (1+z)^{\frac{1}{2}}$, the Inverse Multiquadratic (IMRBF), $\phi(z) = (1+z)^{-\frac{1}{2}}$, and the Cauchy (CRBF), $\phi(z) = (1+z)^{-1}$. In the output layer, the activations of the hidden units are combined in order to produce a classification of the input pattern.

In this study, we investigate the q-Gaussian RBFNN, which can reproduce different RBFs, by changing a real parameter q. A Hybrid Algorithm (HA) is employed to select the parameters of the Radial Basis Functions (RBFs): the number of hidden nodes and the centers, width and the value of the parameter q of each q-Gaussian RBFNN of the population.

This paper is organized as follows: a brief analysis of some works related with the models proposed is given in Section 2; Section 3 describes base classifier

E.S. Corchado Rodriguez et al. (Eds.): HAIS 2010, Part II, LNAI 6077, pp. 280–287, 2010.

applied to binary-classification problems; In section 4, a methodology to optimize the RBF parameters based on Hybrid Algorithms is presented ; Section 5 explains the experiments carried out; and finally, Section 6 summarizes the conclusions of our work.

2 Related Works

A RBFNN is a three-layer feed-forward Neural Network. Let the number of nodes of the input layer, of the hidden layer and of the output layer be p, m and 1 respectively. For any sample $\mathbf{x} = [x_1, x_2, \ldots, x_p]$, the output of the RBFNN is $f(\mathbf{x})$. The model of a RBFNN can be described with the following equation:

$$f(\mathbf{x}) = \beta_0 + \sum_{i=1}^{m} \beta_i \cdot \phi_i(d_i(\mathbf{x})) \tag{1}$$

where $\phi_i(d_i(\mathbf{x}))$ is a non-linear mapping from the input layer to the hidden layer, $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ is the connection weight between the hidden layer and the output layer, β_0 is the bias. The function $d_i(\mathbf{x})$ can be defined as:

$$d_i(\mathbf{x}) = \frac{\|\mathbf{x} - \mathbf{c}_i\|^2}{\theta_i^2} \tag{2}$$

where θ_i is the scalar parameter that defines the width for the i-th radial unit, $\|.\|$ represents the Euclidean norm and $\mathbf{c_i} = [c_1, c_2, \ldots, c_p]$ the centers of the RBFs. The standard RBF (SRBF) is the Gaussian function, which is given by:

$$\phi_i(d_i(\mathbf{x})) = e^{-d_i(\mathbf{x})}, \tag{3}$$

The radial basis function $\phi_i(d_i(\mathbf{x}))$ can take different forms, including the Cauchy RBF (CRBF) defined by:

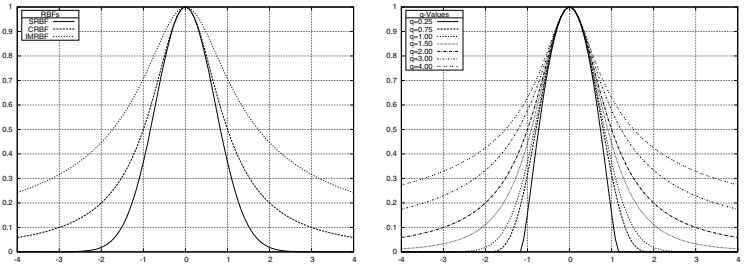
$$\phi_i(d_i(\mathbf{x})) = \frac{1}{1 + d_i(\mathbf{x})} \tag{4}$$

and the Inverse Multiquadratic RBF (IMRBF), given by:

$$\phi_i(d_i(\mathbf{x})) = \frac{1}{(1 + d_i(\mathbf{x}))^{\frac{1}{2}}} \tag{5}$$

Fig. 1 ilustrates the influence of the choice of the RBF in the hidden unit activation. One can observe that the Gaussian function presents a higher activation close to the radial unit center than the other two RBFs. In this paper, we propose the use of the q-Gaussian function as RBF. The q-Gaussian can be defined as:

$$\phi_i(d_i(\mathbf{x})) = \begin{cases} (1 - (1-q)d_i(\mathbf{x}))^{\frac{1}{1-q}} & \text{if } (1 - (1-q)d_i(\mathbf{x})) \geq 0 \\ 0 & \text{Otherwise.} \end{cases} \tag{6}$$



(a) (b)

Fig. 1. Radial unit activation in one-dimensional space with $c = 0$ and $\theta = 1$ for different RBFs: (a) SBRF, CRBF and IMRBF and (b) q-Gaussian with different values of q

The q-Gaussian can reproduce different RBFs for different values of the real parameter q. As an example, when the q parameter is close to 2, the q-Gaussian is the CRBF, for $q = 3$, the activation of a radial unit with an IMRBF for $d_i(\mathbf{x})$ turns out to be equal to the activation of a radial unit with a q-Gaussian RBF for $d_i(\mathbf{x})$ and, finally, when the value of q converges to 1, the q-Gaussian converges to the Gaussian function (SRBF). Fig. 1b presents the radial unit activation for the q-Gaussian RBF for different values of q. As we can see in Fig. 1b, a small change in the value of q represents a smooth modification on the shape of the RBF.

3 q-Gaussian RBF for Classification

To apply evolutionary neural network techniques, we consider a RBFNNs with softmax outputs and the standard structure: an input layer with a node for every input variable; a hidden layer with several RBFs; and an output layer with 1 node. There are no connections between the nodes of a layer and none between the input and output layers either. The activation function of the i-th node in the hidden layer ($\phi_i(d_i(\mathbf{x}))$) is given by Eq. 6 and the activation function of the output node ($f(\mathbf{x})$) is defined in Eq 1. The transfer function of all output nodes is the identity function.

In this work, the outputs of the neurons are interpreted from the point of view of probability through the use of the softmax activation function.

$$g(\mathbf{x}) = \frac{\exp f(\mathbf{x})}{1 + \exp f(\mathbf{x})} \tag{7}$$

where $g(\mathbf{x})$ is the probability that a pattern $\mathbf{x}$ belongs to class 1. The probability a pattern $\mathbf{x}$ has of belonging to class 2 is $1 - g(\mathbf{x})$.

The error surface associated with the model is very convoluted. Thus, the parameters of the RBFNNs are estimated by means of a HA (detailed in Section

1: **Hybrid Algorithm**:
2: Generate a random population of size N
3: **repeat**
4: Calculate the fitness of every individual in the population
5: Rank the individuals with respect to their fitness
6: The best individual is copied into the new population
7: The best 10% of population individuals are replicated and they substitute the worst 10% of individuals
8: Apply parametric mutation to the best (p_m)% of individuals
9: Apply structural mutation to the remaining $(100 - p_m)$% of individuals
10: **until** the stopping criterion is fulfilled
11: Apply *iRprop+* to the best solution obtained by the EA in the last generation.

Fig. 2. Hybrid Algorithm (HA) framework

4). The HA was developed to optimize the error function given by the negative log-likelihood for N observations, which is defined for a classifier g:

$$l(g) = \frac{1}{N} \sum_{n=1}^{N} [-y_n f(\mathbf{x}_n) + \\ + \log \exp f(\mathbf{x}_n)] \,. \tag{8}$$

where y_n is the class that the pattern n belongs to.

4 Hybrid Algorithm

The basic framework of the HA is the following: the search begins with an initial population of RBFNNs and, in each iteration, the population is updated using a population-update algorithm which evolves both its structure and weights. The population is subject to operations of replication and mutation. Figure 2 describes the procedure to select the parameters of the radial units. The main characteristics of the algorithm are the following:

1. *Representation of the Individuals.* The algorithm evolves architectures and connection weights simultaneously, each individual being a fully specified RBFNN. The neural networks are represented using an object-oriented approach and the algorithm deals directly with the RBFNN phenotype.
2. *Error and Fitness Functions.* We consider $l(g)$ (see Eq. 7) as the error function of an individual g of the population. The fitness measure needed for evaluating the individuals is a strictly decreasing transformation of the error function $l(g)$ given by $A(g) = \frac{1}{1+l(g)}$, where $0 < A(g) \leq 1$.
3. *Initialization of the Population.* The initial population is generated trying to obtain RBFNNs with the maximun possible fitness. First, $5,000$ random RBFNNs are generated. The centers of the radial units are firstly defined by the k-means algorithm for different values of k, where $k \in [M_{min}, M_{max}]$, being M_{min} and M_{max} the minimum and maximum number of hidden nodes allowed for any RBFNN model in the HA. The widths of the RBFNNs

are initialized to the geometric mean of the distance to the two nearest neighbourhoods and the q parameter to values near to 1, since when $q \to 1$ the q-Gaussian reduces to the standard Gaussian RBFNN. A random value in the $[-I, I]$ interval is assigned for the weights between the hidden layer and the output layer. The obtained individuals are evaluated using the fitness function and the initial population is finally obtained by selecting the best 500 RBFNNs.

4. *Structural Mutation.* Structural mutation implies a modification in the structure of the RBFNNs and allows the exploration of different regions in the search space, helping to keep the diversity of the population. There are four different structural mutations: hidden node addition, hidden node deletion, connection addition and connection deletion. These four mutations are applied sequentially to each network, each one with a specific probability. If the structural mutator adds a new node in the RBFNN, the q parameter is assigned to a γ value, where $\gamma \in [0.75, 1.25]$, since when $q \to 1$ the q-Gaussian reproduce to the SRBF.
5. *Parametric Mutation.* Different weight mutations are applied:
   - *Centre, Radii and q Mutation.* These parameters are modified in the following way:
     - Centre creep. The value of each centre is modified by adding a Gaussian noise, $c_{ji}(t+1) = c_{ji}(t) + \xi(t)$, where $\xi(t) \in N(c_{ji}, r_i)$ and $N(c_{ji}, r_i)$ represents a one-dimensional normally distributed random variable with mean c_{ji} and with variance the radius of the RBF hidden node.
     - Radius creep. The value of each radii is modified by adding another Gaussian noise, $r_i(t+1) = r_i(t) + \xi(t)$, where $\xi(t) \in N(r_i, d)$ and $N(r_i, d)$ represents a one-dimensional normally distributed random variable with mean r_i and with variance the width of the range of each dimension (d).
     - Mutation of the q parameter. The q parameter is updated by adding an ε value, where $\varepsilon \in [-0.25, 0.25]$, since the modification of the q-Gaussian RBFNN is very sensible to q variation (as we can see in Fig. 1b).
   - *Output-to-Hidden Node Connection Mutations* [3]. These connections are modified by adding another Gaussian noise, $w(t+1) = w(t) + \xi(t)$, where $\xi(t) \in N(0, T(g))$ and $N(0, T(g))$ represents a one-dimensional normally distributed random variable with mean 0 and variance the network temperature ($T(g) = 1 - A(g)$).
6. *iRprop+ Local Optimizer.* The local optimization algorithm used in our paper is the *iRprop+* [6] optimization method. The *iRprop+* is believed to be a fast and robust learning algorithm. This algorithm applies a backtracking strategy (i.e. it decides whether to take a step back along a weight direction or not by means of a heuristic). In the proposed methodology, we run the EA and then apply the local optimization algorithm to the best solution obtained by the EA in the last generation.

Table 1. Characteristics of the eleven datasets used for the experiments: number of instances (Size), number of Real (R), Binary (B) and Nominal (N) input variables, total number of inputs (#In.), number of classes (#Out.), per-class distribution of the instances (Distribution), minimum and maximum number of hidden nodes used for each dataset ($[M_{\min}, M_{\max}]$) and the number of generations (#Gen.)

Dataset	Size	R	B	N	In	Out	Distribution	$[M_{\min}, M_{\max}]$	*Gen*
Labor	57	8	3	5	29	2	(30, 27)	[2, 5]	20
Promoters	106	–	–	57	114	2	(53, 53)	[2, 5]	100
Hepatitis	155	6	13	–	19	2	(32, 123)	[2, 5]	20
Sonar	208	60	–	–	60	2	(98, 110)	[2, 5]	40
Heart	270	13	–	–	13	2	(150, 120)	[2, 5]	100
BreastC	286	4	3	2	15	2	(201, 85)	[2, 5]	40
Heart-C	302	6	3	4	26	2	(164, 138)	[2, 5]	100
Liver	345	6	–	–	6	2	(145, 200)	[2, 5]	40
Vote	435	–	16	–	16	2	(267, 168)	[2, 5]	20
Card	690	6	4	5	51	2	(307, 308)	[4, 7]	40
German	1000	6	3	11	61	2	(700, 300)	[1, 3]	200

All nominal variables are transformed to binary variables.
BreastC: Breast-Cancer; Heart-C: Heart-disease (Cleveland).

5 Experiments

5.1 Experimental Design

The proposed methodologies are applied to eleven datasets taken from the UCI repository [7], to test its overall performance when compared to other radial basis functions (SRBF, CRBF and the Inverse Multiquadratic RBF (IMRBF)). The selected datasets present different numbers of instances and features (see Table 1).

The experimental design was conducted using a 10-fold cross validation, with 10 repetitions per each fold. The performance of each method has been evaluated using the correct classification rate in the generalization set (C_G).

All the parameters used in the evolutionary algorithm except the maximun and minimun number of RBFs in the hidden layer and the number of generations have the same values in all problems analyzed below (see Table 1). We have done a simple linear rescaling of the input variables in the interval $[-2, 2]$, X_i^* being the transformed variables. The connections between hidden and output layer are initialized in the $[-5, 5]$ interval (i.e. $[-I, I] = [-5, 5]$). The size of the population is $N = 500$. For the structural mutation, the number of nodes that can be added or removed is within the $[1, 2]$ interval, and the number of connections to add or delete in the hidden and the output layer during structural mutations is within the $[1, 7]$ interval.

Table 2. Comparison of the proposed basis functions to other basis functions: Mean and Standard Deviation (SD) of the accuracy results ($C_{\mathrm{G}}(\%)$) from 100 executions, mean accuracy ($\overline{C}_{\mathrm{G}}(\%)$), mean ranking ($\overline{R}$), p-Value and $\alpha^{'}$ for the Hommel post-hoc non-parametric tests in C_G with $\alpha = 0.1$ (q-Gaussian is the control method)

	Method($C_{\mathrm{G}}(\%)$)			
	SRBF	CRBF	IMRBF	q-Gaussian
Labor	91.33 ± 12.09	$\mathbf{95.00 \pm 11.24}$	91.66 ± 8.78	*93.33 ± 11.65*
Promoters	75.54 ± 13.56	80.18 ± 6.66	*81.09 ± 8.69*	$\mathbf{84.00 \pm 6.15}$
Hepatitis	$\mathbf{86.33 \pm 8.09}$	83.16 ± 7.15	85.12 ± 7.52	*85.30 ± 7.54*
Sonar	$\mathbf{78.38 \pm 9.03}$	74.09 ± 10.20	76.02 ± 11.16	*76.04 ± 13.56*
Heart	81.85 ± 8.97	83.70 ± 8.76	$\mathbf{84.81 \pm 8.45}$	*84.07 ± 7.20*
BreastC	72.04 ± 6.39	71.35 ± 8.00	$\mathbf{73.10 \pm 6.39}$	*73.06 ± 6.77*
Heart-C	85.44 ± 3.83	85.45 ± 5.59	*85.77 ± 3.05*	$\mathbf{85.79 \pm 5.20}$
Liver	*68.41 ± 5.15*	65.23 ± 8.23	65.52 ± 6.31	$\mathbf{71.30 \pm 6.50}$
Vote	$\mathbf{96.32 \pm 3.97}$	95.39 ± 3.59	94.94 ± 2.36	*96.08 ± 3.45*
Card	86.08 ± 3.14	*86.52 ± 3.55*	85.94 ± 3.80	$\mathbf{87.87 \pm 0.37}$
German	74.80 ± 3.82	*74.90 ± 3.17*	74.40 ± 2.50	$\mathbf{75.25 \pm 2.98}$
$\overline{C}_{\mathrm{G}}(\%)$	81.50	81.36	*81.67*	**82.91**
$\overline{R}$	*2.72*	2.99	2.72	**1.54**
p-Value	0.03	0.00	0.03	-
$\alpha^{'}_{Hommel}$	0.10	0.03	0.05	-

The best result is in bold face and the second best result in italics.

5.2 Comparison to Other Radial Basis Functions

In Table 2, the mean and the standard deviation of the correct classification rate in the generalization set (C_G) is shown for each dataset and a total of 100 executions. From the analysis of the results, it can be concluded, from a purely descriptive point of view, that the q-Gaussian model obtained the best results for five datasets, the SRBF achieved the best performace for three dataset and the CRBF and IMRBF methods yield the higher performance for one and two datasets respectively.

To determine the statistical significance of the rank differences observed for each method in the different datasets, we have carried out a non-parametric Friedman test [8] with the ranking of C_{G} of the best models as the test variable (since a previous evaluation of the C_{G} values results in rejecting the normality and the equality of variances' hypothesis). The test shows that the effect of the method used for classification is statistically significant at a significance level of 10%, as the confidence interval is $C_0 = (0, F_{0.10} = 2.89)$ and the F-distribution statistical values are $F^* = 8.34 \notin C_0$ for C_{G}. Consequently, we reject the null-hypothesis stating that all algorithms perform equally in mean ranking.

Based on this rejection, the Hommel post-hoc test is used to compare all classifier to each other. The Hommel test was applied with the best performing model (q-Gaussian) as the control method. The results of the Hommel tests for $\alpha = 0.10$ can be seen in Table 2, using the corresponding p and $\alpha^{'}_{Hommel}$ values.

From the results of these tests, it can be concluded that the q-Gaussian model obtains a significantly higher ranking of C_{G} when compared to the remaining of RBFs, which justifies the proposal.

6 Conclusions

In this paper, we have proposed a new approach to determine the optimized parameters for the q-Gaussian RBF applied to multi classification problems. These models have been designed with a HA constructed specifically to take into account the characteristics of this kernel model. The evaluation of the model and the algorithm for the eleven datasets considered, showed that the q-Gaussian RBF obtained a higher accuracy when compared to the remaining RBFs. Finally, some suggestions for future research are the following: to study other radial basis functions and to adapt the algorithm to deal with multi-class problems.

Acknowledgement

This work has been partially subsidized by the TIN 2008-06681-C06-03 project of the Spanish Inter-Ministerial Commission of Science and Technology (MICYT), FEDER funds and the P08-TIC-3745 project of the "Junta de Andalucía" (Spain). The research of Francisco Fernández-Navarro has been funded by the "Junta de Andalucia" Predoctoral Program, grant reference P08-TIC-3745.

References

1. Lippmann, R.: Pattern classification using neural networks. IEEE Communications Magazine 27, 47–64 (1989)
2. Bishop, C.M.: Neural networks for pattern recognition. Oxford University Press, Oxford (1996)
3. Martínez-Estudillo, F.J., Hervás-Martínez, C., Gutiérrez, P.A., Martínez-Estudillo, A.C.: Evolutionary product-unit neural networks classifiers. Neurocomputing 72(1-2), 548–561 (2008)
4. Gutiérrez, P.A., Hervás-Martínez, C., Carbonero, M., Fernández, J.C.: Combined Projection and Kernel Basis Functions for Classification in Evolutionary Neural Networks. Neurocomputing 72(13-15), 2731–2742 (2009)
5. Freeman, J.A.S., Saad, D.: Learning and generalization in radial basis function networks. Neural Computation 7(5), 1000–1020 (1995)
6. Igel, C., Hüsken, M.: Empirical evaluation of the improved rprop learning algorithms. Neurocomputing 50(6), 105–123 (2003)
7. Asuncion, A., Newman, D.: UCI machine learning repository (2007)
8. Friedman, M.: A comparison of alternative tests of significance for the problem of m rankings. Annals of Mathematical Statistics 11(1), 86–92 (1940)