GA-RBF: A Self-Optimising RBF Network

Ben Burdsall
Mastère 2IA
ENST de Bretagne, BP 832
29285 Brest Cedex, France
burdsall@gti.enst-bretagne.fr

Christophe Giraud-Carrier
Department of Computer Science
University of Bristol
Bristol, BS8 1UB, England
cgc@cs.bris.ac.uk

Abstract

The effects of a neural network's topology on its performance are well known, yet the question of finding optimal configurations automatically remains largely open. This paper proposes a solution to this problem for RBF networks. A self-optimising approach, driven by an evolutionary strategy, is taken. The algorithm uses output information and a computationally efficient approximation of RBF networks to optimise the K-means clustering process by co-evolving the two determinant parameters of the network's layout: the number of centroids and the centroids' positions. Empirical results demonstrate promise.

1 Introduction

Radial basis function (RBF) networks (e.g., [8, 9, 13]) are a class of hybrid connectionist models. Whilst they are essentially three-layer feedforward networks, RBF networks differ from classical multilayer perceptrons in three significant ways: there is only one set of trainable weights, from the hidden layer to the output layer; the nodes' activation functions are non-standard (i.e., neither *sign* nor a sigmoid); and learning is effected by both supervised and unsupervised techniques.

In a RBF network, the nodes of the hidden layer encode a set of well positioned centroids, each representing one or part of a class. Generally, the centroids are obtained by K-means clustering (i.e., unsupervised learning), whilst the weights of the output layer are trained by a single-shot process using pseudo-inverse matrices or SVD (i.e., supervised learning). RBF networks have found wide applicability in traditional classification problems as well as in modern fuzzy control systems.

As with other neural network models, experience shows that the performance of RBF networks is greatly affected by their topology, that is, the choice of centroids making up the hidden layer. Too many centroids leads to over-fitting, while too few centroids may prove insufficient to capture intrinsic class divisions adequately. In general, the network's classification accuracy is influenced primarily by the number of centroids used to represent each class and the position of each centroid within its class.

With K-means, the number of centroids per class is set a priori by the user and is typically the same for all classes. Whilst this greatly simplifies the design of RBF networks and is probably adequate for "regular" problem spaces, it is often too restrictive and generally impractical. In complex problem spaces with multiple classes, it is likely that, though many classes consist of only a few compact clusters, other classes may exhibit singularities that require additional centroids. In a hyper-dimensional problem space, such singularities cannot be detected readily by a human user. Another difficulty with K-means is its *random* choice of starting positions for the centroids. Such non-determinism hinders robustness and directly impacts performance. Finally, K-means defines clusters for one class at a time and thus does not take into account potential interactions between classes.

A variety of techniques have been suggested to overcome the limitations of the traditional, K-means-based approach to centroid selection. They range from algebraic solutions inspired by wavelet transforms [10] to symbolic solutions using dynamic regression trees [2] and concept learning algorithms [1] to statistical solutions using means tracking [14], clustering [1] and simulated annealing [6] to genetically-based solutions [12] to more ad hoc techniques such as elimination [4]. The solution proposed here, like [12], is based on the use of an evolutionary strategy for self-optimisation. Unlike [12], where the genetic algorithm (GA) evolves functionally-equivalent canonical parametrisations

of RBF networks, the GA here evolves fuzzy centroids that become the basis functions of the network's hidden layer. Both the number of centroids per class and their positions are optimised. Because the GA acts on all of the classes at once and measures fitness as classification accuracy, the method naturally profits from global information about class interactions. Furthermore, robustness is increased by eliminating the non-deterministic choice of starting positions.

The paper is organised as follows. Section 2 describes GA-RBF, a self-optimising RBF network. Section 3 presents empirical results on the performance of GA-RBF against traditional RBF networks and related models. Section 4 summarises and concludes the paper.

2 GA-RBF

To construct an optimal network layout, GA-RBF uses output information and a computationally efficient approximation of RBF networks to optimise K-means clustering by co-evolving the two determinant parameters of the network's topology: the number of centroids and the centroids' positions.

2.1 Encoding

A real-valued, rather than a binary encoded, genetic algorithm (GA) is used, causing the representation to be of a more phenotypical nature. Each chromosome, or individual, encodes one set of centroids starting positions for each class, as follows.

$$individual \stackrel{\text{def}}{=} [\{s_1^1, \ldots, s_{k_1}^1\}, \ldots, \{s_1^n, \ldots, s_{k_n}^n\}]$$

where n is the number of classes and k_i is the number of centroids for class i. The k_i's vary continually during evolution, thus providing the necessary diversity in the population to optimise both determinant parameters. In the current implementation, a practical upper bound of 7 is placed on the k_i's since empirical evidence suggests that most problems require fewer than 7 centroids per class.

2.2 Fitness

GA-RBF's training set is split into a clustering set and an evaluation set. The objective fitness function used for evaluating individuals consists of the application of the K-means algorithm to the clustering set using the starting positions encoded in the individual's genes, followed by a test classification of the evaluation set using the K-means-computed centroids and the nearest-attracting prototype (NAP) classifier [3]. Hence,

$$fitness \stackrel{\text{def}}{=} \frac{correct\_classifications}{size\_of\_evaluation\_set} \times 100$$

The NAP classifier extends the classical nearest-neighbour classifier by using infinite fuzzy support as in RBF networks. Using NAP classifiers rather than RBF networks saves the construction of the networks and the associated computation of the output weights at each generation of the GA. Hence, the NAP classifier is used here as a computationally efficient approximation of a RBF network's performance. Note that although the centroids are labelled during optimisation, they are not labelled once placed in the RBF network upon convergence. The early labelling allows GA-RBF to use output information and NAP, to conduct a faster, more informed search for an optimal solution.

2.3 Reproduction

The selection of individuals for reproduction is biased towards fitter individuals, as follows. Let N be the size of the population and $fitness(i)$ denote the fitness of individual i. The probability that i is selected is:

$$p_i = \frac{fitness(i)}{\sum_{j=1}^{N} fitness(j)}$$

The p_i's define a probability density function over the population. The parents chosen for reproduction are obtained by sampling the distribution without replacement. Genetic operators are applied to pairs of parents to produce offsprings that replace the least fit individuals in the population.

Because a phenotypic representation is used, the traditional genetic operators have to be modified to maintain the integrity of the solution space. The classical crossover operator is replaced by a novel operator, called *gene-pooling*, and three different types of mutation are introduced.

Gene-pooling takes two parents and produces a single offspring. Following [5], a population is viewed as a pool of genes rather than a collection of individuals. A gene, here, is a single centroid's starting position. During crossover the genes of two parents are mixed together into a kind of genetic soup. From this mixture, a new individual is formed by random selection of genes. The chromosome size of the offspring is determined by a random number which follows a normal distribution

centred on the average parental chromosome size. Non-individuals are eliminated by constraining the mixture of genes to contain at most one instance of each gene. Gene-pooling is a valid form of crossover since, in the adopted representation, the order of genes on a chromosome is irrelevant.

Mutation is applied to offsprings and consists of adding, removing or swapping a centroid in a randomly chosen class in the chromosome. Let K be the maximum number of genes per class (here, $K = 7$). Let i be the randomly selected class and k_i be the number of genes in class i. The probabilities of occurrence of each mutation are given by:

$$\begin{aligned} P_{add} &\stackrel{\text{def}}{=} \frac{k_i}{K} \times \gamma_{add} \\ P_{remove} &\stackrel{\text{def}}{=} \frac{K-k_i}{K} \times \gamma_{remove} \\ P_{swap} &\stackrel{\text{def}}{=} 0.3 \end{aligned}$$

where $\gamma_{add} = 0.67$ and $\gamma_{remove} = 0.875$. Mutation by adding and mutation by removing are generally mutually exclusive. Together with gene-pooling, they provide a way of optimising the number of centroids per class. The centroids added or swapped during mutation are drawn at random from within the clustering data points for the selected class. They may not already exist in the class.

2.4 Population Control

In addition to the above, a punishment function is used to combat over-fitting and an aging function is used to prevent saturation by super-individuals.

Individuals with more centroids than others generally have higher fitness as they can place their numerous centroids near the training examples. Unfortunately, these individuals also tend to overspecialise on the evaluation set used and to perform poorly on other sets, i.e., over-fitting occurs. To prevent over-fitting, GA-RBF allows individuals to evolve larger number of centroids per class but punishes them for doing so. Here, the punishment introduces a form of natural handicap for highly fit, greedy individuals. It is given by:

$$punishment \stackrel{\text{def}}{=} \sum_{i=1}^{n} \alpha_i \times \frac{1.38}{n} \times (k_i - 3) \times fitness(i)$$

where n, k_i and $fitness(i)$ are as before, and α_i is 1 if $k_i > 3$ and 0 otherwise. Only greedy individuals are punished and their punishment increases with both greed and fitness. With the addition of the punishment mechanism, the actual fitness function for GA-RBF becomes:

$$fitness' \stackrel{\text{def}}{=} fitness - punishment$$

Individuals whose fitness far exceed the average fitness of the population are termed super-individuals. Super-individuals reproduce rapidly and come to dominate the population after only a few generations. The propagation of super-individuals leads to rapid exploitation but often results in premature convergence to a local optimum. A reasonable amount of exploration must be maintained. To address this issue, the notion of a life span for individuals has been suggested (see, [7]). GA-RBF follows this idea and defines the life span of individual i, at "birth", to be:

$$Life\_span(i) \stackrel{\text{def}}{=} \beta \times \frac{fitness(i)}{Mean}$$

where $Mean$ is the average fitness of the population and β is some constant. Each time an individual reproduces, its life span is reduced by 1. When the life span reaches 0 the individual dies and is removed from the population. Should a parent die following reproduction, its offspring takes its place rather than that of the weakest individual in the population.

2.5 Network Construction

The GA population is initialised by creating between 30 and 60 random individuals. Starting positions for each class are chosen at random from among that class' data points in the clustering set. The GA population is then evolved until convergence. Upon convergence, the starting positions encoded in the fittest individual are used by K-means to compute the centroids which are then fed into the hidden layer of the RBF network. Finally, the output layer of the network is trained using SVD.

3 Experimental Results

Two datasets from [11] were used to test GA-RBF. The noisy versions contain the original data with substantial noise added. Multiple simulations and cross-validation techniques were used to increase the validity of the results. The results for GA-RBF are compared against those of a NAP classifier with 3 (K-means-generated) centroids per class, an user-optimised infinite RBF network with 3 centroids per class, and an optimised multilayer perceptron (MLP). Results are in Table 1 and ecord predictive accuracy on unseen test data. The fixed number 3 of centroids per class used in the non-evolutionary models results from intensive, user-driven testing. With GA-RBF, the number of centroids per class

Table 1: Experimental Results

Application	GA-RBF	RBF	MLP	NAP
Iris	96.3	96.3	96.0	95.6
Iris (N)	84.4	81.7	83.0	79.9
Diabetes	95.6	95.1	96.0	95.2
Diabetes (N)	89.6	89.0	85.1	86.3
Averages	**91.5**	**90.5**	**87.5**	**89.3**

evolves naturally to settle on a value between 2 and 4. The cost of the evolution process is relatively low as convergence occurs in less than 3 minutes (Sun Workstation). Moreover, with no a priori knowledge nor manual fine tuning, GA-RBF results in a slight increase of predictive accuracy for the problems considered.

4 Conclusion

This paper presents GA-RBF, a self-optimising RBF network. GA-RBF harnesses the power of genetic algorithms to optimise the K-means clustering process by co-evolving the determinant parameters of the network's topology. GA-RBF uses a real-valued genetic encoding and custom-designed genetic operators. With its fitness based on classification accuracy, GA-RBF also enhances the search with output information. Empirical results show that GA-RBF is a powerful optimiser.

Because it is such a strong optimiser, the GA itself tends to lead to over-fitting. This is particularly problematic with small datasets. In the experiments reported here, the training set was split only once into clustering and evaluation sets. Improved performance could be expected with cross-validation during optimisation. Alternatively, one could consider extending the fitness function with anti-over-fitting.

Acknowledgements

This work was supported in part by the DES Defence Engineering and Science Group, UK.

References

[1] C. Baroglio, *et al.* Learning Controllers for Industrial Robots. *Machine Learning*, 23(2/3):221-249, 1996.

[2] E. Blanzieri, *et al.* Growing Radial Basis Function Networks. In *Proceedings of the Fourth Workshop on Learning Robots*, 1995.

[3] B. Burdsall and C. Giraud-Carrier. Evolving Fuzzy Prototypes for Efficient Data Clustering. In *Proceedings of ISFL'97*, 1997.

[4] C. Decaestecker. Design of a Neural Net Classifier Using Prototypes. Technical Report, IRIDIA, Université Libre de Bruxelles, 1994.

[5] L.J. Fogel. Evolutionary Programming in Perspective: The Top Down View. In J.M. Zurada, *et al.* (Eds.), *Computational Intelligence: Imitating Life*, IEEE Press, Inc., NY, 1994.

[6] B. Lemarié and A-G. Debroise. A Dynamical Architecture for Radial Basis Function Networks. In *Proceedings of ICANNGA'95*, 305-308, 1995.

[7] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer Verlag, 1992.

[8] J. Moody and C. Darken. Learning with Localized Receptive Fields. In *Proceedings of the 1988 Connectionist Models Summer School*, Pittsburgh, PA, 1988.

[9] J. Moody and C. Darken. Fast Learning in Networks of Locally-Tuned Processing Units. *Neural Computation*, 1(2):281-294, 1989.

[10] S. Mukherjee and S. Nayar. Automatic Generation of RBF Networks. Technical Report CUCS-001-95, Columbia University, Department of Computer Science, 1995.

[11] P.M. Murphy and D.W. Aha. UCI Repository of Machine Learning Databases. University of California, Irvine, Department of Information and Computer Science.

[12] R. Neruda. Functional Equivalence and Genetic Learning. In *Proceedings of ICANNGA'95*, 53-56, 1995.

[13] T. Poggio and F. Girosi. Networks for Approximation and Learning. *Proceedings of IEEE*, 78(9):1481-1497, 1990.

[14] K. Warwick, *et al.* Center Selection for a Radial Basis Function Network. In *Proceedings of ICANNA'95*, 309-312, 1995.