Neural Comput & Applic (2004) 13: 193–201
DOI 10.1007/s00521-004-0404-5

ORIGINAL ARTICLE

C. Harpham · C. W. Dawson · M. R. Brown

A review of genetic algorithms applied to training radial basis function networks

Received: 24 February 2004 / Accepted: 24 February 2004 / Published online: 24 April 2004

Abstract The problems associated with training feedforward artificial neural networks (ANNs) such as the multilayer perceptron (MLP) network and radial basis function (RBF) network have been well documented. The solutions to these problems have inspired a considerable amount of research, one particular area being the application of evolutionary search algorithms such as the genetic algorithm (GA). To date, the vast majority of GA solutions have been aimed at the MLP network. This paper begins with a brief overview of feedforward ANNs and GAs followed by a review of the current state of research in applying evolutionary techniques to training RBF networks.

Keywords Artificial neural network · Genetic algorithm · Multilayer perceptron · Radial basis function

1 Introduction

Feedforward artificial neural networks (ANNs) maintain a high level of research interest due to their ability to map any function to an arbitrary degree of accuracy. This has been demonstrated theoretically for both the radial basis function (RBF) network [1] and the popular multilayer perceptron (MLP) network [2]. Due to their similarity in functional mapping ability, the RBF and MLP networks share the same problem domains and consequently direct comparisons have been made [3]. Feedforward ANNs have been applied to many diverse areas such as pattern recognition, time series prediction, signal processing, control and a variety of mathematical applications.

During the late 1960s, the linear mapping limitations of the single layer perceptron were theoretically demonstrated by Minsky and Papert [4] resulting in a dramatic decline of research in this field. During this period a suitable algorithm for training multilayer perceptrons, which enables nonlinear functions to be mapped, had yet to be discovered. The popularising of the backpropagation training algorithm by Rumelhart and McClelland [5], which enabled multilayer networks to be trained, stimulated a renewal of interest in perceptron networks during the 1980s.

In contrast, the RBF network [6, 7] was developed from an exact multivariate function interpolation [8] and has attracted a lot of interest since its conception. There are a number of significant differences between RBF and MLP networks:

- The RBF network has one hidden layer and the MLP network has one or more hidden layers.
- The hidden and output layer nodes of the RBF network are different while the MLP network nodes are usually the same throughout.
- RBF networks are locally tuned while MLP networks construct a global function approximation.

This paper looks at training feedforward ANNs (specifically RBF networks) using GA techniques and provides a literature review of the current state of research of applying GAs to the RBF network.

2 The RBF network

The RBF network consists typically of two layers (see Fig. 1), with architecture similar to that of a two layer MLP network. The distance between an input vector

C. Harpham (✉)
Department of Geography,
King's College London, Strand, London,
WC2R 2LS, UK
E-mail: colin.harpham@kcl.ac.uk

C. W. Dawson
Department of Computer Science,
Loughborough University,
Leicestershire, LE11 3TU, UK

M. R. Brown
Department of Computing,
University of Central Lancashire, Preston,
Lancashire, PR1 2HE, UK

and a prototype vector determines the activation of the hidden layer with the nonlinearity provided by the basis function. The nodes in the output layer usually perform an ordinary linear weighted sum of these activations, although nonlinear output nodes are an option.

Mathematically the network output is expressed by the following:

$$y_k(\mathbf{x}) = \sum_{j=1}^{M} w_{kj} F_j(\mathbf{x}) + w_{k0} \tag{1}$$

where x is the d-dimensional input vector with elements

x_i

and w_{kj} are the final layer weights and w_{k0} is the bias.

The basis function $F_j(\mathbf{x})$ for the popular Gaussian function is expressed as

$$F_j(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{u}_j\|^2}{2\sigma_j^2}\right) \tag{2}$$

where σ is the width of the basis function and u_j is the vector determining the centre of the basis function F_j with elements $u_{ji.}$

Training an RBF network with linear outputs is accomplished in two stages. The first stage is unsupervised and accomplished by obtaining cluster centres of the training set input vectors. A popular method is k means clustering first applied by Moody and Darken [7] although other methods such as the Max-Min [9] or Kohonen network [10] can be employed. The second stage consists of solving a set of linear equations the solution of which can be obtained by a matrix inversion technique such as singular value decomposition (SVD) or by least squares.

3 The genetic algorithm

The pioneering work of Holland [11] illustrated how the Darwinian evolution process can be applied, in the form of an algorithm, to solve a wide variety of problems. Due to the biological motivation this highly parallel adaptive system is now called the genetic algorithm (GA). The GA has a population of individuals competing against each other in relation to a measure of fitness, with some individuals breeding, others dying off, and new individuals arising through combination and mutation.

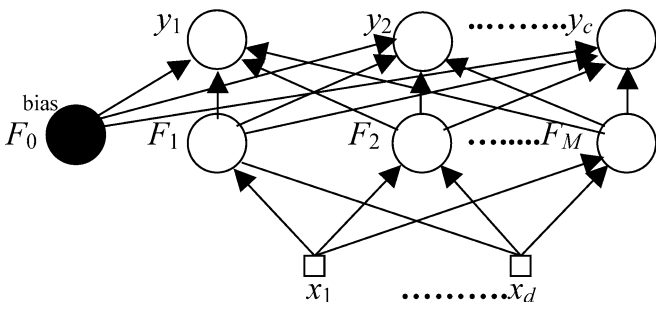

Fig. 1 Typical RBF network configuration

3.1 An overview of the GA

The GA in its simplest form utilises fixed length character strings and the steps in the algorithm based on [12] are as follows:

1. Randomly create an initial population of individual character strings with probability 0.5 of each bit being a 0 or 1.
2. Assign a fitness value to each individual in the population using the fitness measure.
3. Create a new population by applying *reproduction*, *crossover* and *mutation* operations to the individual strings. These genetic operations are applied to chosen strings in the population with a probability based on fitness.
   a) Reproduce an existing individual by copying it into the new population.
   b) Create two new strings from two existing strings by using the crossover operation at a randomly chosen starting point.
   c) Create a new string from an existing one by randomly mutating a character.
   d) Evaluate the fitness of the new population.
4. Has the convergence criteria been reached? If not go to step 3.
5. The string that has given the optimum value is designated as the result of the GA run. Convergence is not achieved in the usual sense since there is always an element of mutation, which reintroduces an individual string to the search space.

3.2 The genetic operators

3.2.1 Reproduction

Two mates are chosen from the population on the basis of their fitness values; the fitter strings having a higher probability of entering the mating pool and consequently the weaker ones a higher probability of dying off. Firstly, the fitness values have to be mapped to positive values and these values inverted for minimisation purposes, thus giving the highest probability to the lowest value. The function obtains, through roulette wheel selection, the pairs of mates. The roulette wheel is biased towards the fitter strings as it is partitioned according to the ratio of the fitness values divided by the sum of the fitness values. Consequently, the higher the fitness value the greater the chance of being selected.

3.2.2 Crossover

A point in the string for the pair of mates is randomly selected and binary characters after this point are swapped between the two.e.g.

$S_1 = 110110|01011$
$S_2 = 010100|01110$
$S'_1 = 11011001110$
$S'_2 = 01010001011$

where S_1, S_2 are parent strings and S'_1, S'_2 are the resulting offspring.

3.2.3 Mutation

Mutation provides a mechanism for introducing new material into the gene pool, thus preventing the algorithm from getting stuck in local minima. The strings are mutated in a bit by bit process. However, the probability of mutation is usually set very low and if selected the binary character is swapped from 0 to 1 or vice versa.

3.3 Setting the parameters

Studies of GAs for function optimisation [13] have indicated that good performance requires a high probability of crossover, a low probability of mutation and a moderate population size. Typical values for crossover are in the range 0.4–0.9 and to calculate the probability of mutation the empirically derived formula [14] can be used as a starting point:

$$Pmutation \approx \frac{1}{N\sqrt{L}} \tag{3}$$

where N is the number of generations and L is the string length. The algorithm described above can, despite its simplicity, cope with quite complex problems. The most significant departure from this approach is to use a higher cardinality representation for the strings, typically base 10 [15]. This can be particularly convenient when integer representation is required over a range that does not have a one to one mapping with base 2, (bearing in mind the upper limit must be all ones). If a large amount of information needs to be encoded into the strings and they consequently become excessively long, more advanced multipoint crossover operators need to be considered. For an introductory text on GAs see Coley [16] and for a more in-depth coverage see Goldberg [12], Michalewicz [15] and Back [17].

4 Training problems

There are two problem areas associated with training feedforward ANNs: firstly, determining the optimal architecture and secondly, determining the optimal weights (or basis centres and weights in the case of the RBF network).

4.1 The architecture

To avoid laborious trial and error, techniques have been developed for both network types for overcoming this problem, although the MLP network has attracted far more attention. The main approaches are destructive (frequently referred to as pruning) and constructive methods. A destructive method commences with a sufficiently large network and then prunes nodes and/or connections until an optimal network is obtained. A constructive method commences with a minimal network adding nodes and layers as required.

The MLP and RBF networks have a common problem in how many hidden nodes to allocate. Too few and a network fails to learn, too many and its ability to generalise is poor (often referred to as overtraining).

4.1.1 MLP networks

The importance of obtaining a minimal (i.e. optimal) network is reflected by the large amount of research interest received. The variations on the theme of pruning [18] are considerable, although the main drawback is determining a sufficiently large network to begin with. Constructive methods [19] have not attracted the same level of attention as pruning. The most popular example is *cascade correlation* [20] where nodes and layers are added until an optimal network is obtained. Unfortunately, the network produced is generally too complex and improved performance is obtained by applying pruning [21].

4.1.2 RBF networks

The problem of network complexity is somewhat alleviated in the case of the RBF network since there is only one hidden layer and the number of data vectors in the training set defines an upper bound for the number of hidden nodes.

In the work by Chen et al. [22] the number of hidden layer nodes is determined by utilising Akaike's information criterion [23], which provides a compromise between network complexity and network performance. An error reduction ratio is used to select the basis centres, although error reduction is local so the network may become trapped in local minima. Chen's network was trained using the orthogonal least squares algorithm.

The network proposed by Holcomb and Morari [24] starts with one hidden layer node and adds nodes as necessary. The locations of the hidden nodes are optimised using a commercial optimisation package. However, the termination criterion is not clear and may be problem dependent.

Lee and Kil [25] developed a hierarchically self-organising learning algorithm. This algorithm starts with no hidden nodes and adds them based upon whether the input vector lies within a defined boundary; if not, a node is added. An alternative approach, proposed by Musavi et al. [26], begins with as many nodes as data vectors, progressively reducing them by the use of an iterative clustering algorithm. Since with these methods the number of nodes and the locations are determined by clustering the input vectors, there is no guarantee that they will produce an optimal network.

Sundararajan et al. [27] provide a review of this area including a proposed solution for sequential learning, which they refer to as a ''minimal resource allocation network''.

4.2 Weight optimisation

4.2.1 MLP networks

A popular algorithm for training an MLP network is *back propagation*, which begins with the weights initialised to random values. The algorithm minimises an error term by using gradient descent; consequently, the starting point in weight space may result in convergence to a local minima rather than the global one. Using the most basic form of the algorithm the problem can be reduced by using different starting points in weight space for each run of a particular network configuration. This makes training the network a very slow and laborious task and could easily result in a sub-optimal network. There have been many proposed methods for addressing this problem including the use of optimisation algorithms such as *simulated annealing* [28] where the stochastic element prevents convergence to local minima.

4.2.2 RBF networks

The performance of the RBF network is dependant upon the basis centres, which must be representative of the whole data set. The popular K means algorithm does have a number of problems associated with it. The number of hidden nodes required, i.e. cluster centres, has to be decided a priori. Two samples close to each other in the input space do not necessarily have similar outputs. The random choice of starting point influences the final cluster centres and so can only achieve a local optimal solution. Due to these shortcomings, the poor performance of an RBF network is likely to be attributed to sub-optimal placement of the cluster centres.

5 GAs for training ANNs

There are several general reviews of this area [29, 30, 31, 32] and further relevant reading is provided by [33, 34, 35]. A high proportion of this literature is aimed at the MLP network; however, many of the ideas and methods are directly relevant to RBF networks. The following taxonomy presents the main areas where GAs have been applied to feedforward ANNs:

5.1 Evolving network architecture

The GA can be used to optimise the number of layers, nodes and the connections between them. Applying a GA to evolving MLP architecture involves addressing the same problem as pruning in that an upper bound must be defined on the network complexity.

5.2 Determining the connection weights

By using a GA to optimise the connection weights the possibility of converging to a poor local minima discussed earlier is eliminated and, while there is no guarantee of finding the global minimum, the stochastic nature of the algorithm will have thoroughly explored the search space. Since the GA returns a near-optimal solution, a popular hybrid technique is applying gradient descent after the GA has found a near optimal solution in order to home in on that local minima.

5.3 Evolving learning parameters

The application of a GA to the optimisation of a network's learning rules and parameters provides the ANN with the ability to dynamically adapt to suit its architecture and application. This area has greater relevance to MLP networks since the standard RBF has only one algorithm parameter, the basis width.

5.4 The optimisation of the data set

A GA can also be applied to finding an optimal subset of the training data, which is then used for training the network [36]. However, this topic lies outside the scope of this review.

5.5 An application to MLP networks

In the case of the MLP applied to problems of reasonable complexity the GA has to simultaneously optimise all unknown parameters and the network weights creating an extremely large search space for the GA to optimise. The larger the problem the greater the population and number of generations required to ensure that the GA has had sufficient time to optimise the search space. Furthermore, the larger the number of parameters to be encoded the greater the chromosome length, resulting in the need for more complex multi-point crossover techniques being required in order for the GA to create sufficient diversity. This is entirely problem-

dependent and care must be exercised not to overload the GA resulting in sub-optimal performance. One solution is *indirect coding* where the string encodes the architecture and learning parameters for the network. The amount of information encoded into each string is then greatly reduced with the consequent reduction in the search space for the GA. The disadvantage of this approach is that a network must be trained for each string in the population in order to compute the fitness.

In the case of optimising the architecture, the same problem encountered with pruning arises in that an upper bound must be given on the number of hidden nodes and layers to prevent over fitting of the training data.

5.6 An application to RBF networks

Because the RBF network has only one hidden layer the task of applying the GA algorithm for optimising the architecture is simplified. Furthermore, the number of data vectors in the training set defines an upper bound on the number of hidden nodes. As discussed earlier optimally determining the basis centres is problematical since all methods have their pitfalls; also, determining the number of basis centres is generally accomplished by trial and error. Therefore, this encourages a hybrid approach where the GA optimises the basis centres and architecture and the second stage continues to utilise a supervised training method (using, for example, singular value decomposition). Consequently, many of the articles reviewed here adopt this approach.

6 A review of GA/RBF literature

This section overviews the current state of research of applying GAs to the RBF network.

6.1 A search for an optimal subset and/or an optimal architecture

A popular application of GAs to RBF training is in the search for a subset of input vectors to provide optimal basis centres. This has significant advantages over searching for the centres in $\mathbf{R}^n$ because the search space is finite and discrete. There is, however, the ''parameter'' or ''competing conventions'' problem to resolve [37]. This is where two networks may be functionally equivalent, except the nodes together with matching weights, are in a different order. If the network is directly encoded onto the string then a standard crossover procedure can easily create such a network.

An indirect coding method presented by Whitehead and Choate [38] evolves space-filling curves to distribute radial basis functions over an input space. This is used in conjunction with gradient-based learning to determine the weights. Binary coded strings are used to represent the parameters for an algorithm that generates the basis function centres. The gradient-based learning rate and space filling curve parameters are encoded onto the genotype using an eight-bit representation. The crossover operator has a 50% chance of falling within the boundary of the eight-bit encoded parameters, otherwise, bit level recombination is permitted to avoid the ''blocking problem'' [39]. The fewer the number of hidden nodes the more training passes are allowed, thus improving the fitness and biasing the training procedure to a minimal network. When applied to the Mackey-Glass [40] time series, better generalisation capabilities are reported than obtained with a network using k-means clustering and gradient based learning.

In Whitehead and Choate [41] an unusual approach is proposed where instead of each string representing a network, the whole GA population represents one network. Each string in the population encodes an RBF and width that is part of the same network. In order to provide a fitness evaluation, credit is assigned to each RBF based on its contribution to the overall prediction of the network. The problem referred to as ''niching'' in the GA literature [42] is addressed by varying the degree of competition between RBFs based upon their degree of overlap. The outputs weights are determined by SVD. When applied to the Mackey-Glass time series, improved results are reported compared to the k means algorithm. Further tests on a pattern recognition problem were inconclusive.

Whitehead [43] develops ideas from Whitehead and Choate [41] with the significant difference that orthogonal niche based credit apportionment is utilised. When applied to the Mackey glass time series, better predictive performance is reported than with RBF networks produced by the orthogonal least squares method and by k means clustering.

Billings and Zheng [44] used a GA to automatically configure a network by finding an optimal subset n_c from the N_t training set examples. Each network is coded as a variable length string using distinct integer representation for the data points

$$x_i(i = 1, 2, \ldots, N)$$

They propose two crossover operators, one of fixed length that preserves the length of the parent string and one of variable length that changes the string length. Their work using this representation and genetic operators was based on methods proposed by Lucasius and Kateman [45]. In order to provide a compromise between network complexity and network performance, their formulation of an objective function to minimise incorporates Akaike's [23] information criterion. The method was tested on a pattern recognition problem with promising results.

Neruda [46] investigates functionally equivalent networks [47] for the case of radial basis function networks. These ideas provide the basis for a proposed genetic learning rule that operates on a small part of the whole weight space. Restricting the search space to one

parameterisation for each class overcomes the "competing conventions" problem described earlier. For this so called "canonical parameterisation", custom genetic operators have been proposed. There are no experimental results given.

The approach proposed by Carse et al. [48] makes use of genetic operators developed for fuzzy logic systems [49]. Some motivation for the work had been provided by the similarity between RBF networks and certain types of fuzzy rule-based systems demonstrated by Jang and Sun [50]. The crossover operator minimises disruption of "schemata" composed of hidden layer nodes with significantly overlapping basis functions and avoids the duplication of hidden layer nodes. The hybrid approach employed, determines the fitness by training the network using the least mean square algorithm. The method is applied to a simple mathematical function with promising results.

Neruda's reasoning inspired Carse and Fogarty [51, 52] to extend their GA/RBF work to incorporate canonical parameterisation. The encoding and genetic operators used are developed from Carse et al. [48]. The approach was successfully applied to the Mackey Glass time series.

Burdsall and Giraud-Carrier [53] propose a self-optimising classifier RBF network using fuzzy prototypes. They used real value representation with a variable string length to determine both the number of hidden nodes and the cluster centres. Since each class may require several fuzzy centroids, the optimal number of classes also evolves. Instead of using the RBF network for evaluating the fitness function, computation time is reduced by using a nearest-attracting prototype classifier [54] that gives an efficient approximation to the RBF network's performance.

In Kuncheva [55] a direct binary coding method is used with each string representing a subset of the training data. The binary representation indicates the presence (1) or absence (0) of a data vector and is initialised randomly with probability 0.5. Mating uses the entire population set in order to maintain genetic diversity and avoid premature convergence and a uniform crossover operator is employed [56, 57]. The work is aimed at classification problems with experimental results given for the IRIS and two-spirals problems. The GA/RBF approach did not improve on a 1-NN classifier for the IRIS data (probably attributed to the small sample size) although a significant improvement is reported for the two spirals data set.

A hybrid neural structure proposed by Chaiyaratana and Zalzala [58] is composed of one RBF network and one MLP network for each RBF network output. The GA is used to determine the optimal basis centres for the RBF part of the hybrid. The MLPs of the hybrid network have their weights optimised by a variation of *back propagation*. The two spirals classification problem is used to compare the hybrid method with an RBF network trained using least squares algorithm. An increase in efficiency is reported when using the hybrid approach.

Sergeev et al. [59] use a GA technique to minimise the network hidden layer and the pattern set required for training. The binary encoded chromosome comprises of identical length segments, the number of which corresponds to the number of hidden nodes in the network. The first gene in the chromosome indicates whether there is a node specified by this segment, followed by genes, which encode the basis function width and finally the identification number of a potential basis centre. The scheme is applied to training a dynamical object emulator with promising results.

In Xue and Watton [60] the GA/RBF is applied to the dynamics modelling of fluid power systems. They apply a global error descent (GED) algorithm to a self-organising radial basis function network. The network employs a constructive approach, growing the number of hidden nodes. The GA is used as a first stage to select the initial basis centres from the training data and increases the number of hidden nodes until the network overfits. The topology of the network having been established, the GED is then applied to adjust the centres and weights until the decreasing error is not significant. The weights are then renewed by the least squares algorithm and the GED is applied again. If the process meets a local minimum the GA is applied again to provide a new search direction.

In Vesin et al. [61] a GA is used to determine the basis centres for the RBF. They first randomly select M candidate centres from the set of input vectors, thus defining the chromosome as an M-bit binary string. To compute the fitness the chromosome is decoded and least squares used to calculate the output weights. To avoid over-complex networks the MDL criterion [62] is implemented, which does not, however, take into account the cost of coding widths and centres to avoid over-penalising. When applied to the sunspots benchmark series [63] the LSE was found to be superior to an RBF trained using the orthogonal least squares algorithm [64].

Moechtar et al. [65] use the GA/RBF network approach for power-system transient stability evaluation. The GA is utilised to find the optimal basis centres and is a fairly conventional binary encoded string. The orthogonal least squares algorithm determines the output layer weights. The authors report satisfactory results using this method although no direct comparisons are made.

In Dawson et al. [66] the GA/RBF is applied to the prediction of Ranunculus in the rivers Test and Itchen. The GA uses binary encoded fixed length strings with standard genetic operations for evolving the basis centres. The second layer weights were determined by SVD with the output error of the network on the training data providing the fitness value. The GA/RBF performed slightly better than a RBF network using k means and slightly worse than a MLP network.

As part of an investigation into data mining using a variety of techniques, Sumathi et al. [67] apply the GA to optimising the basis centres and output weights of a RBF network. The network parameters were encoded

onto a binary chromosome where a matrix crossover is used for reproduction and a stochastic remainder used for selection. When applied to data mining through classification for two problems, namely image segmentation and heart disease data, the RBF network showed a marked improvement when trained with a GA.

Leung et al. [68] report that they used the GA RBF to employ the GA to search for the optimum centres, variance and the number of hidden nodes. The parameters are encoded onto the chromosome as real numbers and following Billings and Zheng [44] the network complexity is controlled by an approximation of Akaike's information criterion [23]. The technique is applied to the detection of small objects in sea clutter. A previous investigation using an RBF network [69] gave no improvement on a linear model; however, the GA-RBF captured the clutter nonlinearity giving a significant improvement.

6.2 Optimising all parameters

Sheta and De Jong [70] utilise the GA to optimise the basis centres, basis function width and output weights; however, no precise details are given as to the encoding method. A comparison is made with an RBF trained using LSE when applied to the time-series forecasting of currency (GBP/dollar) exchange rate, where their results prove to be very similar. The authors suggest this is due to the lack of noise and abrupt changes in the data.

6.3 Optimising network learning parameters only

Chen et al [71] propose a two level learning approach where at the lower level the network is trained using a regularized orthogonal least squares algorithm. Regularisation is a technique for improving the generalisation performance of the network. At a higher level the GA optimises the two key learning parameters, the regularisation parameter and the hidden node widths. This method has been developed from an earlier work [72]. The two parameters are each coded into 16 bit strings with a population size of five. The number of crossover points was set to four without any mutation. The method is applied successfully to a simple scalar function with added noise, prediction of the Mackey-Glass time series, and sunspot time series prediction.

6.4 A modular approach

In Jiang et al. [73] a structural modular neural network is implemented, which was inspired by a proposal of Lui's [74], where the hidden layer consists of both sigmoidal and Gaussian transfer functions. This combined MLP and RBF network has the number of nodes optimised by a genetic algorithm together with, in the case of the RBF, their centres. The chromosome in the GA's population is divided into two sections, the first part is binary and of fixed length and encodes the number of hidden MLP nodes, the second section is real valued and of variable length and encodes the number and position of the RBF hidden nodes. The individual chromosomes are decoded and trained using the Levenburg-Marquardt algorithm [75], the MSE on the training data providing the fitness value for the GA to optimise. The scheme is applied to the strength modelling of concrete under triaxial stresses where favourable results are reported in comparison to standard MLP and RBF networks.

6.5 Using a GA as a clustering algorithm

An alternative approach to finding an optimal subset proposed by Aiguo and Jiren [76] uses a GA as a clustering algorithm to obtain the RBF centres to overcome the potential pitfalls of the k-means clustering algorithm mentioned earlier. They propose two approaches.

6.5.1 1st method

Suppose m samples in the training set can be partitioned into n clusters. A cluster solution is then coded as a string $S = \{s_1, s_2, \ldots s_i \ldots s_m\}$ where s_i is a l bit binary coding. If $s_i = k$ and $k \leqslant n$, then the ith sample belongs to the kth cluster.

If $s_i = k$ and k, then the ith sample belongs to the (k-n)th cluster.

6.5.2 2nd method

The locations of the RBF centres are coded as a string

$$S = \{s_{11}, \ldots, s_{1d}, s_{21}, \ldots, s_{2d}, s_{n1}, \ldots, s_{nd}\}$$
$$\text{where } \{s_{i1}, \ldots, s_{id}\}(i = 1, \ldots, n)$$

represents the location of the ith centre and s_{ij} is a l' bit binary coding.

The basis centre widths are calculated from the variances of the clusters and the final layer weights are determined using the recursive least squares (RLS) algorithm. The method is tested on a simulated two-dimensional system and the Mackey-Glass time series. Improved predictive accuracy compared with K means clustering is reported.

7 Conclusions

While the majority GA ANN training has so far has focussed on the MLP, the GA also has potential benefits when applied to RBF training. To date, RBF training using GAs has focussed on determining the optimal subset of data inputs for the basis centres and estab-

lishing how many are required. The selection of an optimal subset is of paramount importance since this provides the foundation for good RBF network performance. From the literature reviewed here, the GA has been shown to be superior to clustering algorithms.

The number of basis centres could still be found by trial and error, however, most of the papers reviewed here incorporate variable string lengths to determine the number of basis centres. Unfortunately, this potentially creates a problem of overfitting the training data. A common solution reported is to apply a penalty term to the fitness evaluation, which increases with the number of nodes. Another suggestion is to incorporate a validation set in the fitness calculation. Consequently, a worthwhile topic for further research would be to establish conclusively the best approach to solve the problem of over fitting.

Another potential area for further research is in applying the GA/RBF network to "real world" problems. This has not received a lot of attention so far and further investigation is required in order to provide a truer evaluation of the approach.

References

1. Park J, Sandberg IW (1991) Universal approximation using radial basis function networks. Neur Comput 3(2):246–257
2. Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. Neur Netw 2:359–366
3. Niranjan M, Fallside F (1998) Neural networks and radial basis functions in classifying static speech patterns. Report CUED/FINFENG/TR22, University Engineering Department, Cambridge University, UK
4. Minsky ML, Papert SA (1990) Perceptrons. MIT Press, Cambridge, MA
5. Rumelhart DE, McClelland JL (eds) (1986) Parallel distributed processing: explorations in the microstructure of cognition, vol 1. MIT Press, Cambridge, MA
6. Broomhead DS, Lowe D (1988) Multivariable function interpolation and adaptive networks. Comp Sys 2:321–355
7. Moody JE, Darken CJ (1989) Fast learning in networks of locally-tuned processing units. Neur Comput 1(2):281–294
8. Powell MJD (1987) Radial basis functions for multivariable interpolation: a review. In: Mason JC, Cox MG (eds) Algorithms for approximation, Clarendon Press, Oxford, UK
9. Batchelor BG, Wilkins BR (1969) Method for location of clusters of patterns to initialise a learning machine. Elect Lett 5:481–483
10. Kohonen T (1982) Self-organised formation of topologically correct feature maps. Biol Cybern 43:59–69
11. Holland J (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor, MI
12. Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley, Reading, MA
13. De Jong KA (1975) An analysis of the behaviour of a class of genetic adaptive systems. Dissertation, University of Michigan
14. Schaffer JD, Caruana RA, Eshelman LJ, Das R (1989) A study of control parameters affecting online performance of genetic algorithms for function optimisation. In: Schaffer JD (ed) Proceedings of the 11th International Joint Conference on Artificial Intelligence, Morgan Kaufmann, San Mateo, CA
15. Michalewicz Z (1996) Genetic algorithms + data structures = evolution programs, 3rd edition. Springer, Berlin Heidelberg New York
16. Coley DA (1999) An introduction to genetic algorithms for engineers and scientists. World Scientific, New York
17. Back T (1996) Evolutionary algorithms in theory and practice. Oxford University Press, New York
18. Reed R (1993) Pruning algorithms—a survey. IEEE Trans Neur Netw 4:740–746
19. Kwok TY, Yeung Y (1997) Constructive algorithms for structure learning in feedforward neural networks for regression problems. IEEE Trans Neur Netw 8(3):630–645
20. Fahlman SE, Lebiere C (1990) The cascade-correlation learning architecture. In: Touretzky DS (ed) Advances in neural information processing systems, vol 2, Morgan Kaufmann, Los Altos, CA
21. Chudova DI, Dolenko SA, Orlov YuV, Pavlov DYu, Persiantsev IG (1998) Benchmarking of different modifications of the cascade correlation algorithm. In: Parmee I (ed) Proceedings of the 3rd International Conference on Adaptive Computing in Design and Manufacture. Springer, Berlin Heidelberg New York
22. Chen S, Billings SA, Cowan CFN, Grant PW (1990) Practical identification of narmax models using radial basis functions. Int J Contr 52(6):1327–1350
23. Akaike H (1973) Information theory and an extension of the maximum likelihood principle. In: Petrov BN, Csaki F (eds) Proceedings of the 2nd International Symposium on Information Theory, Tsahkadsov, Armenia
24. Holcomb T, Morari M (1991) Local training for radial basis function networks: towards solving the hidden unit problem. In: Proceedings of the American Control Conference, Boston, MA, June 1991
25. Lee S, Kil RM (1991) A Gaussian potential function network with hierarchically self-organising learning. Neur Netw 4:207–224
26. Musavi MT, Ahmed W, Chan KH, Faris KB, Hummels DM (1992) On the training of radial basis function classifiers. Neur Netw 5:595–603
27. Sundararajan N, Saratchandran P, Lu Ying Wei (1999) Radial basis function neural networks with sequential learning: MRAN and its applications. World Scientific, New York
28. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimisation by simulated annealing. Science 220(4598):671–680
29. Schaffer JD, Whitley D, Eschleman LJ (1992) Combinations of genetic algorithms and neural networks: a survey of the state of the art. In: Proceedings of the International Workshop on Combinations of Genetic Algorithms and neural Networks (COGANN-92), IEEE Computer Society Press, Los Alamitos, CA
30. Yao X (1993) A review of evolutionary artificial neural networks. Int J Intellig Sys 8(4):539–567
31. Yao X (1999) Evolving artificial neural networks. Proc IEEE 87(9):1423–1447
32. Balakrishnan K, Honavar V (1995) Evolutionary design of neural architecture—a preliminary taxonomy and guide to literature. Technical report, AI Research Group, CS-TR 95–01
33. Whitley D (1995) Genetic algorithms and neural networks. In: Winter G, Periaux J, Galan M, Cuesta P (eds) Genetic algorithms in engineering and computer science, Wiley, New York
34. Masters T (1993) Practical neural network recipes in C++. Academic Press, New York
35. Rooiji AJF, Jain LC, Johnson RP (1997) Neural network training using genetic algorithms. Series in machine perception and artificial intelligence, vol. 26, World Scientific Publishing, New York
36. Reeves CR, Taylor SJ (1998) Selection of training data for neural networks by a genetic algorithm. Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York
37. Hancock PJB (1992) Genetic algorithms and permutation problems: a comparison of recombination operators for neural net structure specification. In: Schaffer JD, Whitley D. and Eschleman LJ (eds) Proceedings of the International Workshop on Combinations of Genetic Algorithms and Neural Networks, (COGANN-92), IEEE Computer Society Press, Los Alamitos, CA

38. Whitehead BA, Choate DC (1994) Evolving space-filling curves to distribute radial basis functions over an input space. IEEE Trans Neur Netw 5(1):15–23
39. Goldberg DE (1991) Real coded genetic algorithms, virtual alphabets and blocking. Compl Sys 5:139–167
40. Mackey MC, Glass L (1977) Oscillation and chaos in physiological control systems. Science 197:287–289
41. Whitehead BA, Choate DC (1996) Cooperative-competitive genetic evolution of radial basis function centres and widths for time series prediction. IEEE Trans Neur Netw 7:869–880
42. Deb K, Goldberg DE (1989) An investigation of niche and species formation in genetic function optimization. In: Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Francisco
43. Whitehead BA (1996) Genetic evolution of radial basis function coverage using orthogonal niches. IEEE Trans Neur Netw 7(6):1525–1528
44. Billings SA, Zheng GL (1995) Radial basis function network configuration using genetic algorithms. Neur Netw 8(6):877–890
45. Lucasius CB, Kateman G (1992) Towards solving subset selection problems with the aid of the genetic algorithm. In: Manner R, Manderick B (eds) Parallel problem solving from nature, vol 2, Elsevier, Amsterdam
46. Neruda R (1995) Functional equivalence and genetic learning of RBF networks. In: Pearson DW, Steele NC, Albrecht RF (eds) Artificial neural networks and genetic algorithms, Springer, Berlin Heidelberg New York
47. Hecht-Nielsen R (1990) On the algebraic structure of feedforward network weight spaces. In: Advanced neural computers, Elsevier, Amsterdam
48. Carse B, Pipe AG, Fogarty TC, Hill T (1995) Evolving radial basis function neural networks using a genetic algorithm. In: Proceedings of the 2nd IEEE International Conference on Evolutionary Computation, Perth, Australia, 29 November-1 December 1995
49. Carse B, Fogarty TC, Munro A (1996) Evolving fuzzy rule based controllers using genetic algorithms. Int J Fuzz Set Sys 80(3):273–293
50. Jang JR, Sun T (1993) Functional equivalence between radial basis function networks and fuzzy inference systems. IEEE Trans Neur Netw 4:156–159
51. Carse B, Fogarty TC (1996) Tackling the ''curse of dimensionality'' of radial basis function networks using a genetic algorithm. Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York
52. Carse B, Fogarty TC (1996) Fast evolutionary learning of minimal radial basis function neural networks using a genetic algorithm. Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York
53. Burdsall B, Giraud-Carrier C (1997) GA-RBF: a self optimising RBF network. In: Proceedings of the Third International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA'97), Springer, Berlin Heidelberg New York
54. Burdsall B, Giraud-Carrier C (1997) Evolving fuzzy prototypes for efficient data clustering. In: Proceedings of the Second International ICSC Symposium on Fuzzy Logic and Applications (ISFL'97), ICSC Academic Press, New York
55. Kuncheva LI (1997) Initializing of an RBF network by a genetic algorithm. Neurocomput 14(3):273–288
56. Beasley D, Bull DR, Martin RR (1993) An overview of genetic algorithms: Part 1, fundamentals. Univer Comput 15:59–69
57. Beasley D, Bull DR, Martin RR (1993) An overview of genetic algorithms: Part 2, research topics. Univer Comput 15:170–181
58. Chaiyaratana N, Zalzala AMS (1998) Evolving hybrid RBF-MLP networks using combined genetic/unsupervised/supervised learning. In: Proceedings of the UKACC International Conference on CONTROL '98, Swansea, UK, September 1998
59. Sergeev SA, Mahotilo KV, Voronovsky GK, Petrashev SN (1998) Genetic algorithm for training dynamical object emulator based on RBF neural network. Int J Appl Electro Mech 9(1): 65–74
60. Xue Y, Watton J (1998) Dynamics modelling of fluid power systems applying a global error descent algorithm to a self-organising radial basis function network. Mechatronics 8(7): 727–745
61. Vesin JM, Gruter R (1999) Model selection using a simplex reproduction genetic algorithm. Sig Process (78):321–327
62. Rissanen J (1989) Stochastic complexity in statistical enquiry, World Scientific, Singapore
63. Tong H (1990) Non-linear time series, Oxford University Press, Oxford, UK
64. Chen S, Cowan CFN, Grant PM (1991) Orthogonal least squares learning algorithm for radial basis function networks. IEEE Trans Neur Netw 2(2):302–309
65. Moechtar M, Farag AS, Hu L, Cheng TC (1999) Combined genetic algorithms and neural network approach for power-system transient stability evaluation. Europ Trans Elect Power 9(2):115–122
66. Dawson CW, Wilby RL, Harpham C, Brown MR, Cranston, E, Darby EJ (2000) Modelling Ranunculus presence in the Rivers test and Itchen using artificial neural networks. The Fifth International Conference on GeoComputation, Greenwich, UK, August 2000
67. Sumathi S, Sivanandam SN, Ravindran R (2001) Design of a soft computing hybrid model classifier for data mining applications. Engin Intell Sys Electr Engin Comm 9(1):33–56
68. Leung H, Dubash N. Xie N (2002) Detection of small objects in clutter using a GA-RBF neural network. IEEE Trans Aero Electr Sys 38(1): 98–118
69. Leung H, Lo T (1993) Chaotic radar signal processing over the sea. IEEE J Ocean Eng 18:287–295
70. Sheta AF, De Jong K (2001) Time-series forecasting using GA-tuned radial basis functions. Info Sci 133(3–4):221–228
71. Chen S, Wu Y, Luk BL (1999) Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks. IEEE Trans Neur Netw 10(5):1239–1243
72. Chen S, Wu Y, Alkadhimi K (1995) A two-layer learning method for radial basis function networks using combined genetic and regularised OLS algorithms. In: Proceedings of the 1st IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, Piscataway, NJ, September 1995
73. Jiang N, Zhao ZY, Ren LQ (2003) Design of structural modular neural networks with genetic algorithm. Adv Eng Soft (1):17–24
74. Liu Y, Yao X (1996) Evolutionary design of artificial neural networks with different nodes, evolutionary computation. Proc IEEE Int Conf 1996, 670–675
75. Mor JJ (1977) The Levenberg-Marquardt algorithm: implementation and theory. In: Watson, GA (ed) Numerical analysis, Lecture Notes in Mathematics, Springer, Berlin Heidelberg New York
76. Aiguo S, Jiren L (1998) Evolving Gaussian RBF network for nonlinear time series modelling and prediction. Electr Lett 34(12):1241–1243