ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 341-346

Learning Fault Tolerance in Radial Basis Function Networks

Xavier Parra and Andreu Català

Department of Automatic Control – Technical University of Catalonia
Av. Víctor Balaguer, s/n – 08800 Vilanova i la Geltrú – Barcelona – Spain
Xavier.Parra@upc.es and Andreu.Catala@upc.es

Abstract. This paper describes a method of supervised learning based on forward selection branching. This method improves fault tolerance by means of combining information related to generalization performance and fault tolerance. The method presented focuses on the evolutive nature of the learning algorithm of Radial Basis Function Networks and employs optimization techniques to control the balance between the approximation error with and without faults. The technique developed is empirically analyzed and provides a simple and efficient means of learning fault tolerance. This is illustrated by examples taken from different classification and function approximation problems.

1. Introduction

In general, the main objective of any kind of learning usually is knowledge acquisition. The attempt to reach this aim requires the use of some kind of experience, combined with an empirical knowledge of the problem about which we try to acquire knowledge (this is the case of supervised learning). For example, when considering this latter kind of learning, it is normal to control the knowledge acquisition by means of some sort of optimization strategy or different measures of the error. As an alternative, it is common to use the deviation between the learned and the known outputs or the output predictions or even the evaluation of the generalization capacity in the face of new inputs. Basically, the objective is to get an artificial neural network with acquired knowledge and good generalization. However, knowledge does not have to be restricted to the improvement of a single quality, as could be the generalization capacity of the neural network. The robustness of that neural network or its reliability or fault tolerance are some of the key topics we could consider too when the goal is to acquire knowledge.

Obviously, to tolerate faults that affect structural elements is completely different than to tolerate noise that affects inputs. The former is fault tolerance. The latter is robustness to noisy inputs. The work presented in this paper particularly concentrates on aspects related with fault tolerance and learning. The reasons of having fault tolerant neural systems are diverse. Our interest is in hardware implementation. We are interested on analog implementations (where defects could be found) and on digital implementations (where fault tolerance could suppose a reduction in the number of bits needed to represent the architecture). Initially, fault tolerance, known as the ability of a system to operate correctly in presence of faults, was considered as an inherent quality of artificial neural systems. Later on, it has been shown that fault tolerance is definitely not

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 341-346

an inherent and natural quality of neural systems, though it can be introduced to such systems by different optimization strategies (see [1] and [2]).

Learning capacity of artificial neural networks is the main quality exploited to get better networks as far as fault tolerance is concerned. Training with faults [3] and retraining [4] are two particular techniques of the neural systems, and both are oriented towards the attenuation of the system's degradation in presence of faults. Training with faults is a method based on the modification of the normal training conditions. Techniques based on this method can be split into two major groups. In the first group, we find those techniques that modify some of the conditions in the system's environment but without affecting the learning algorithm [3]. In the second group of learning-with-faults techniques, we find those which actively modify the learning algorithm introducing fault tolerance as a factor to optimize.

Evolutive learning algorithms are of especial interest when trying to introduce essential information about fault tolerance into the learning algorithm with the objective of improving the solution in the sense of being more fault-tolerant. This is the case of radial basis function networks (RBF). The idea of modifying the learning algorithm in order to incorporate significant information related to fault tolerance into it is the main contribution of the work presented.

Learning algorithm and architecture of RBF networks will be briefly described in the following section. In Section 3, we will discuss the method of branching applied during the selection of RBF centers. Section 4 presents results of simulations using one of the techniques developed previously. Finally, we will draw some conclusions regarding the effect of forward selection branching.

2. Learning algorithm of radial basis function networks

Radial basis function networks (RBF) have been traditionally associated with a simple architecture of three layers [2]. Each layer is fully connected to the following one and the hidden layer is composed of a number of nodes with radial activation functions called radial basis functions. Each of the input components feeds forward to the radial functions. The outputs of these functions are linearly combined with weights into the network output. Each radial function has a local response (opposite to the global response of sigmoid function) since their output only depends on the distance of the input from a center point.

Radial functions have a structure that can be represented as follows:

$$\phi_i(\mathbf{x}) = \varphi((\mathbf{x}-\mathbf{c}_i)^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{x}-\mathbf{c}_i)) \tag{1}$$

where φ is the radial function used, $\{\mathbf{c}_i \mid i = 1, 2,\ldots, c\}$ are the radial function centers and R is a metric. The term $(\mathbf{x} - \mathbf{c})^{\mathrm{T}} \mathbf{R}^{-1} (\mathbf{x} - \mathbf{c})$ denotes the distance from the input x to the center c on the metric defined by R. There are several types of functions used, though the Gaussian function is the most typical choice, combined with the Euclidean metric. In this case, the output of the RBF network is:

$$\mathcal{F}(x) = w_0 + \sum_{i=1}^{c} w_i \exp\left(-\frac{\|\mathbf{x}-\mathbf{c}_i\|^2}{r^2}\right) \tag{2}$$

where c is the number of basis functions, $\{w_i \mid i = 1, 2,\ldots, c\}$ are the synaptic weights, $\|\cdot\|$ denotes the Euclidean norm and r is the radius of the radial function.

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 341-346

The RBF learning algorithm is an incremental and evolutionary process. Its mathematical foundation is called subset selection and consists in comparing models made up of different subsets of elements drawn from the same fixed set of candidates. To find the best subset is usually intractable so heuristics must be used to search for a small but hopefully interesting fraction of the space of all subsets. However, the use of these heuristics does not guarantee that the solutions we get include the least number of elements needed to reduce the approximation error to a fixed value.

The heuristic method called forward selection is widely used with RBF networks [5], though it is not the only learning paradigm in RBFs. According to this method, the subset that must be determined is the subset of centers that fix the location of the radial functions in the input space. The method begins with an empty subset to which is added one basis function at a time. The center of the radial function added is selected among the whole set of input patterns and is the one that most reduces the approximation error. The learning process continues until some chosen criterion stops decreasing (e.g. generalized cross-validation).

3. Learning fault tolerance during RBF center selection

Forward selection methods try to minimize the approximation error but do not consider aspects like synaptic weight values or other features such as the size of the network or its fault tolerance. Hence, we can obtain solutions that include large synaptic weight values. Large values for synaptic weights are an unambiguous symptom of low fault tolerance. In general, as it is shown in [6]. Large weight values are likely to correspond to non fault-tolerant candidates. However, this fact makes it possible to enrich the classic optimization strategy used in forward selection methods, in order to consider not only the approximation error, but also the synaptic weight values.

It seems reasonable that if we act properly on the learning in order to incorporate fault tolerance information into it, results should tend to improve this fault tolerance. This assumption can be made since the heuristic used during the progressive selection of radial function centers is not an optimum method.

As described in Section 2, the center of the radial function added during forward selection is the one that most reduces the approximation error. The process starts with an empty set of candidates. The approximation error is assessed for every candidate for the radial function center, supposing that this candidate is included in the network architecture. The process stops with the selection of the candidate that most reduces the approximation error. However, the better candidate according to the approximation error criteria it is not necessarily the better candidate according to the fault tolerance criteria. In fact, often the opposite is the case.

The contribution presented in this paper focus on the center selection which determines the best candidate to be included in the neural architecture. Thus, the basic idea is to determine the error approximation associated to each center candidate and to combine this error with the one obtained when the same candidate for the radial function center is affected by a fault. The new center selection process proposed here still calculates the approximation error under the hypothesis that the center candidate is a part of the neural architecture, as the classic forward selection method does. In addition, the center selection process

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 341-346

calculates the approximation error when the synaptic weight associated to the new radial function, which has the candidate as a center, is affected by a fault.

In order to assess the approximation error in presence of faults, we need to know the laws that characterize these faults. These laws are settled by the fault model. The fault model considered in this work is parametric [7]. Thus, when a neural element faults, its value suffers a variation that linearly depends on a parameter called tolerance parameter (δ). There are at least two possible parametric model laws defined as:

$$\tilde{w} = w + \delta\, w \tag{3}$$

$$\tilde{w} = w + \delta \tag{4}$$

where w is the original value of the neural element and $\tilde{w}$ is the faulty value.

The combination of the approximation error, with and without faults, to generate a single value for each center candidate is made as follows:

$$C = \alpha E + (1-\alpha)T \tag{5}$$

where C is the function to optimize, E is the fault-free approximation error, T is the approximation error with faults and α is a weighting factor. Factor α controls the balance between the approximation error with faults and the approximation error without faults. Thus, when α goes to 1 is equivalent to considering only the fault-free approximation error and this is what we call classic training. On the other hand, when α goes to 0 is equivalent to considering only the approximation error with faults and we shall refer to it as faulty training.

4. Simulations and results

Simulations have been carried out on a real classification problem (Cancer) and a function approximation problem (MacKay's Hermite Polynomial). The classification problem used belongs to PROBEN1 data base and the PROBEN1 standard rules have been used during training [8]. The first partition of the problem has been used, that is to say, 'Cancer1'. The function approximation problem used is from [9] and is based on a one-dimensional Hermite polynomial of equation:

$$y = 1 + (1 - x + 2x^2)e^{-x^2} \tag{6}$$

100 input values are sampled randomly between $-4 < x < 4$ and a Gaussian noise of standard deviation σ=0.1 is added to the outputs. Each data set has been divided into three subsets: training set, validation set and test set, with a relation of 50%, 25% and 25%, respectively. Initially, networks are trained on the training set while the validation set is used to adjust the radial function radius r. Once the radial radius has been fixed, a new training process is carried out. This training process is done on both data sets, the training set and the validation set. The fault model and the fault selection methodology followed are, in both cases, the same model and methodology described in [7].

Table 1 shows results obtained for Cancer and MacKay's Hermite polynomial problems. Although the mean-squared-error over the training set is slightly higher in the faulty case, the errors over the test set are quite similar for classical and faulty training. A remarkable point derived from the detailed study of results is that for both trainings it is possible to get similar learning levels and similar generalization capacities, which makes it possible to say that the general performance for both learning algorithms is equivalent.

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 341-346

Table 1. Training results for Cancer classification problem ($\delta = 0.1$ and $\alpha = 0.1$) and MacKay's Hermite polynomial problem ($\delta = 0.5$ and $\alpha = 0.5$).

Problem	Radius	Classic					Faulty				
		c	Training		Test		c	Training		Test	
			Mean	std	mean	Std		mean	std	Mean	std
Cancer	2.0074	52	0.029	0.092	0.022	0.067	9	0.041	0.113	0.021	0.060
MacKay	0.2041	22	0.009	0.013	0.010	0.015	23	0.009	0.012	0.010	0.014

Table 2. Center selection order for MacKay's Hermite polynomial problem

Step #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	…
Classic	0	119	5	39	12	45	122	63	73	82	147	129	121	52	137	136	…
Faulty	0	119	5	39	12	45	63	122	73	82	129	147	37	48	52	137	…

In Table 2, we find the first seventeen centers selected among the training patterns following the classic and faulty training described above. As can be seen in the data, the set of centers is identical until the twelfth step. At this point, the faulty training allows us to select a branch of centers that diverges from the branch followed with the classic training. From the thirteenth selection, the evolution of classic training has little in common with the evolution of faulty training, since the successive approximation errors strongly depend on the actual set of radial function centers.

Since forward selection branching is proposed as a method of acquiring knowledge about fault tolerance, it seems interesting to compare the fault tolerance of the RBF network we obtain from classic and from faulty training. To do this, we need a measure of the degree of fault tolerance to establish an order between different neural systems. The fault tolerance measure used in this work is called approximation quality [7]. This measure assigns to each neural element a value which is inversely proportional to the degradation generated on the system performance when a fault affects the neural element. In Figure 2, is shown the approximation quality for the Cancer classification problem. We only need to remember that a high value for the approximation quality means good fault tolerance, while a low value means poor fault tolerance. As can be seen, the approximation quality achieved by the faulty training is uniform when the

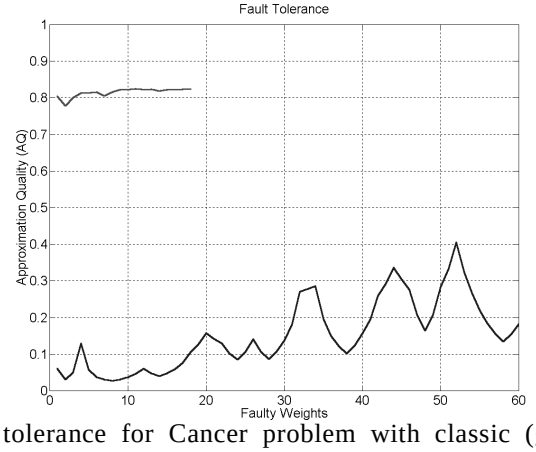

Figure 2: Fault tolerance for Cancer problem with classic (grey) and faulty (black) training.

number of faulty weights grows. When considering the degradation caused by the least fault-tolerant fault (i.e. the left corner in Figure 2), it can be shown that the degradation is 60% worse for the classic training than for the faulty one. Thus, the fault tolerance degree achieved by the faulty training is superior to the one achieved by the classic training.

5. Conclusions

The work presented here shows the possibility of acquiring some kind of knowledge on neural systems fault tolerance. We have shown that fault tolerance is a quality that can be learned by means of an optimization strategy. The adaptation of the learning algorithm to make use of information related with fault tolerance during the training process is particularly interesting when working with RBF networks since their learning algorithm is an evolutive process. This learning quality makes it possible to combine the objective of approximation error minimization with the issue of fault tolerance improvement while the neural system builds its architecture.

Fault tolerance in not an inherent feature of artificial neural networks. Nevertheless, forward selection branching during the selection of RBF centers implemented into the faulty learning method seems to be an efficient technique that, as it supports a similar generalization level, allows us to improve neural systems fault tolerance. Further work remains still to be done to characterize fault tolerance properly and to understand the means by which their fault tolerance can be assured.

References

[1] G.R. Bolt. Fault Tolerance in Artificial Neural Networks. PhD thesis, University York, UK (1992).
[2] D.S. Broomhead and D. Lowe. Multivariable Functional Interpolation and Adaptive Network, Complex Systems, vol. 2, pp. 321-355 (1988).
[3] C.H. Séquin and R.D. Clay. Fault Tolerance in Feedforward Artificial Neural Networks. Neural Networks, vol. 4, pp. 111- 141 (1991).
[4] M.D. Bedworth and D. Lowe, Fault Tolerance in Multi-Layer Perceptrons, RSRE: Pattern Processing and Machine Intelligence Division (1988).
[5] S. Chen, C.F.N. Cowan and P.M. Grant. Orthogonal Least Squares Learning for Radial Basis Function Networks. IEEE Transactions on Neural Networks, vol. 2(2), pp. 302-309 (1991).
[6] X. Parra and A. Català, Sensitivity Analysis of Radial Basis Function Networks for Fault Tolerance Purposes, 5th International Work-Conference on Artificial and Neural Networks (1999).
[7] A. Català and X. Parra, Fault Tolerance Parameter Model of Radial Basis Function Networks. Proceedings of the IEEE International Conference on Neural Networks, pp. 1384-1389 (1996).
[8] L. Prechelt, PROBEN1: Set of Neural Network Benchmark Problems and Benchmarking Rules. Tech. Report 21/94, University of Karlsruhe (1994).
[9] D.J.C. MacKay, Bayesian Interpolation. Neural Computation, vol. 4(3), pp. 415-447 (1992).