International Conference on Neural Networks (ICNN)
Washington, DC June 3-6, 96 pp 199->204

Learning in RBF networks

Michel Verleysen[1] and Kateřina Hlaváčková[2]

[1]Université catholique de Louvain, Microelectronics laboratory
3, pl. du Levant, B-1348 Louvain-la-Neuve, Belgium
phone: +32 10 472551, fax: + 32 10 478667
[2]Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic
verleysen@dice.ucl.ac.be, katka@uivt.cas.cz

ABSTRACT

RBF networks are widely used for the quasi non-parametric estimation of real-valued multi-dimensional functions through a finite set of samples. However, there are numerous ways to estimate the three types of parameters involved in RBF learning: the centers and widths of the radial functions, and their weights. While the literature often deals with the problem of weight computation, many assumptions are usually made on how to choose the center locations and widths of the kernels. This paper shows adaptive ways to evaluate these parameters, based on neural network and vector quantization techniques, and relying on different assumptions on the function to approximate. It also shows the parallel between approximation of functions and of probability densities by RBF networks, and how the techniques developed for one of these domains may be extended to the other one.

1. Introduction

The use of Radial Basis Functions (RBF) for the approximation of real-valued multivariable functions has become more and more popular in the recent literature. RBFs are feed-forward networks that have universal approximation properties (see for example [3]); their main advantage compared to other neural network paradigms is the simplicity of computation of the network parameters. In most methods indeed, the three different types of parameters that can be identified in RBF networks are computed separately (and sequentially), what greatly decreases the complexity of the learning process. In the next sections, we give ways to estimate the three types of parameters, but the main results of this paper concern the important problem of the choice of the widths of the Gaussian kernels used in the network.

2. RBF networks

The problem of interpolation of real multivariable functions can be expressed as follows. Let us consider a set of N data points in the input space $\mathcal{R}^d$, together with their associatied *desired output* values in $\mathcal{R}$:

$$\mathcal{D} = \left\{ (x_i, y_i) \in \Re^d \times \Re, 1 \leq i \leq N \mid f(x_i) = y_i \right\}. \tag{1}$$

This data set can be used to characterize a function with one-dimensional output values; multi-dimensional interpolation can be done by generalizing the following equations and algorithms, while considering separately each component of the output vectors. We consider only one dimensional output in the following.

The RBF approach to approximate function f uses P functions $\phi_j(u) = \phi_j(||u - c_j)||)$, where ϕ_j are radial functions $\phi_j : \Re^+ \rightarrow \Re, 1 \leq j \leq P, u \in \Re^d, c_j \in \Re^d$. The c_j are the locations of the centroids (the centers of the radial basis functions), while $||\cdot||$ denotes a norm on $\Re^n$ (usually Euclidean).

0-7803-3210-5/96 $4.00©1996 IEEE

The approximation of function f may be expressed as a linear combination of the radial basis functions

$$\hat{f}(u) = \sum_{j=1}^{P} \lambda_j \phi\left(\|u - c_j\|\right). \tag{2}$$

The most common radial function in practice is a Gaussian kernel given by

$$\phi_j\left(\|u - c_j\|\right) = e^{-\left(\frac{\|u-c_j\|}{\sigma_j}\right)^2}, \tag{3}$$

where σ_j is the width factor of kernel j.

Once the general shape of the ϕ_j functions is chosen, the purpose of a RBF algorithm is to find parameters c_j, σ_j and λ_j to best fit function f. Fitting means here that the global mean-square error between the desired outputs y_i for all data point $x_i, 1 \leq i \leq N$ and the estimated outputs $\hat{f}(x_i)$ is minimized. This error is given by

$$E_{ms} = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \hat{f}(x_i)\right)^2 = \frac{1}{2}\sum_{i=1}^{N}\left(f(x_i) - \hat{f}(x_i)\right)^2. \tag{4}$$

3. Training of RBF networks

As pointed out in the previous section, training a RBF networks means to find parameters c_j, σ_j and λ_j used in equations 2 and 3 in order to get the best approximation $\hat{f}$ of the function f. In a conventional RBF training (see [2, 1]), these three sets of parameters are succesively and independently computed.

While this splitting of learning into three successive phases is considered as "normal" in most of the literature covering the subject, it must not be forgotten that this method does not find global optimum with respect to all parameters c_j, σ_j and λ_j. However, the splitting makes the learning much easier than in multi-layer perceptrons for example, and this is certainly one of the major advantages of RBF networks. Moreover, it is possible to reach a (local) minimum of the error function 4 with respect to all parameters of the network by using a global gradient descent method which, while loosing the advantages of splitted learning, only requires a few iterations (and usually converges to fairly "good" local minima) if the initial conditions are taken as the result of a splitted learning (see section 3.4.

3.1. Locations of centroids

Locations of centroids are usually chosen according to the probability density of the input set x_i; such choice leads to more centroids, and so naturally to a better approximation of function f, in regions of the input space covered by more input vectors, which seems a good heuristic in many applications.

Moody and Darken [2] proposed to use a k-means clustering algorithm to find the locations of the centroids c_j; as detailed in [1] and [4], we suggest to use a Competitive Learning (CL) method which leads to similar results; with the advantages first of being adaptative (continuous learning, even with an evolving input database), and secondly of helping to the choice of the width factors s explained in the next section. The principle is thus to initialize the P centroids c_j to the first P input patterns x_i, $1 \leq i \leq P$. Next, input vectors x_i, $1 \leq i \leq N$ are sequentially or randomly presented to all centroids c_j, and the centroid c closest to x_i according to

$$\|c_a - x_i\| \leq \|c_j - x_i\|, \forall j \in \{1 \ldots P\}, 1 \leq a \leq P \tag{5}$$

is selected. Then, the selected centroid c_a is moved in the direction of input x_i according to the learning rule

$$c_a(t+1) = c_a(t) + \alpha(t)\|x_j - c_a(t)\|, \tag{6}$$

where $\alpha(t)$ is a time-decreasing adaptation factor, $0 < \alpha(t) < 1$. After convergence of this CL procedure, the probability density of the centroids will approximate the probability density of the input data, making the *regions of influence* of all centroids equiprobable.

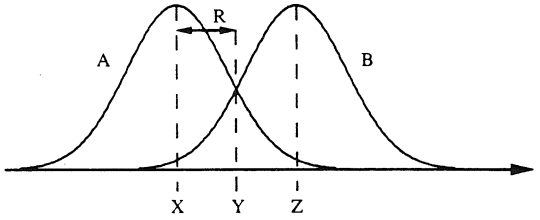

Fig. 1: Illustration of two Gaussian functions in 1 dimension, and related notations.

3.2. Width factors

Moody and Darken [2] proposed to compute the width factors σ_j of the Gaussian RBF units by minimizing a cost function measuring the overlapping between adjacent units; this function however includes a parameter difficult to choose, what effectively thansfers the problem of choosing the σ_j to the choice of this parameter.

In [4] we proposed to use a measure of the dispersion of points in the clusters associated to the centroids in order to fix the width factors σ_j. Keeping the method more general than in [4], an iterative estimation of the width factors can be realized through a simple convex combination between the previous estimation and a new value according to

$$\sigma_a(t+1) = (1-\beta(t))\sigma_a(t) + \beta(t) q \, ||x_i - c_a(t)|| \tag{7}$$

and the width factors σ_a will converge to the mean value of $q \, ||x_i - c_a(t)||$. If q is set to 1, σ_a will converge to the standard deviation of cluster a; the choice of an optimum value for q is explained below.

3.2.1. First case

We first consider that the true probability density of input vectors x_i can be considered as constant on two consecutive clusters, and also that the function f itself can be considered as constant in first approximation on the same range; the first hypothesis leads to the constraint that we will choose the width factor of the Gaussian function associated to each cluster in order to keep the estimate 2 of the density as constant as possible over two consecutive clusters. Let us examine the one-dimensional example of figure 1. X and Z represent the centers of two consecutive clusters A and B, $2R$ the distance and Y the midpoint between them. The purpose of the method is to set the relation between R and the width factor h of the Gaussian functions A and B, in order to have a constant approximate of the probability density over the segment $[X, Z]$. We will simplify the computation of h by setting its value in order to have the same estimate of probability density at points X, Y and Z; we assume that the fluctuations inside the segments $[X, Y]$ and $[Y, Z]$ may be neglected. We will also neglect the influence of a kernel at a distance $2R$ of its center (which, when verified a posteriori, will cause a maximum error of about 6% in the local value of the estimate).

With these hypotheses, we can evaluate the contribution of the Gaussian functions A and B respectively at locations X (or Z) and Y:

$$f(X) = \lambda \tag{8}$$

$$f(Y) = 2\lambda \, e^{\frac{-R^2}{\sigma^2}} \tag{9}$$

The above two equations first suppose that we attributed the same width factor σ to the two clusters centered on X and Z, and secondly that we suppose that the widths factors λ which are respectively associated to these clusters are equal too (which is natural since we consider the function as constant over that range).

Making the estimates 8 and 9 equal to have an equal approximation of probability density at points X, Y and Z leads then to

$$R = \sqrt{ln2}\,\sigma \tag{10}$$

As explained in [4], a similar development which leads to the same result (independent of the dimension of the input space) can be made in dimensions greater than 2.

Now we have the relation between σ and R, we need a method to evaluate R. First, we will evaluate the inertia of each cluster, by using an adaptive method exactly as the competitive learning does for the locations of the centers. The *inertia* coefficient $i(m)$ for each cluster is computed in the following way:

$$i(a) = i(a) + \alpha(||x(n) - c(a)||^2 - i(a)) \tag{11}$$

where a is the index of the closest centroid to a learning vector $x(n)$. Equation 11 is a convex combination at each iteration between the previously estimated value of $i(a)$ and a new contribution $||x(n) - c(a)||^2$ due to the input vector $x(n)$. After learning, parameters $i(m)$, $1 \leq m \leq M_i$, will converge to the average inertia of points in the clusters associated to $c(m)$.

The last point to solve is the relation between the estimated inertia $i(a)$ and the distance R. If we consider that, under the locally uniform density approximation as above, the local arrangement of the centers of consecutive clusters will be as the vertices of an hypercube with edges of length $2R$, the relation between the inertia of each cluster and R is:

$$i(a) = \frac{1}{(2R)^d} \int_V ||x(n) - c(a)||^2 dV = \frac{dR^2}{3} \tag{12}$$

where d is the dimension of the space.

Combining equations 10 and 12 then leads to a width factor σ given, in dimension d, by

$$\sigma = \sqrt{\frac{3\, i(a)}{d\, ln2}} \tag{13}$$

Reminding that σ is adapted according to equation 7, this leads to an optimal value of q given by

$$q = \sqrt{\frac{3}{d\, ln2}} \tag{14}$$

which is the main result of this paper.

3.2.2. Second case

We know consider a less restrictive case, where the density of input vectors x_i can still be considered as constant over two consecutive clusters, but where the function f to approximate is no more constant, but can be linearly approximated over the range of two consecutive clusters.

In this case, it is not natural anymore to consider that the weight factor λ associated to the two consecutive clusters are equal; however, compared to the first case examined above, we can consider that the linear approximation on function f only influences the computation of the width factors λ proportionally to the slope of function f, without any other influence on the computation of the centers and widths of the Gaussian kernels.

It can easily be verified that adding the linear hypothesis on f instead the constant hypothesis in equations 8 and 9 does not influence the result of equation 10. The main result of equation 14 is thus still valid too.

Let us mention however that the above development is no more correct is the function f to approximate greatly differs from its linear approximation on two sonsecutive clusters; rather than being a limitation on the functions that can be approximated by this method, this fixes an upper bound (or at least an order of magnitude) on the distance between two consecutive clusters or, in other words, a lower bound on the number of clusters.

3.2.3. Third case

Let us consider even less restrictive case at last. We make now the hypothesis that both the function f and the density of input vectors x_i can be linearly approximated on two consecutive clusters. This hypothesis is not far from what can be found in real cases: again, such hypothesis fixes a lower bound on the number

of clusters (or Gaussian kernels) that must be used in the approximation, rather than being a limitation on the function f itself.

The only relation where the density of points intervenes is equation 12 which must be changed into

$$i(a) = \frac{1}{(2R)^d} \int_V \|x(n) - c(a)\|^2 p(x) dV \tag{15}$$

where $p(x)$ is the density of points x_n in the cluster. However if this density is linear and if $c(a0$ is the center of teh cluster, the previous result

$$i(a) = \frac{dR^2}{3} \tag{16}$$

Moreover, as in the second case, the multiplication of the Gaussian kernel outputs by different λ_i weight factors does not influence the computation of the centers and widths of the kernels. Finally, this multiplication also does not influence the location of the borders between the clusters (which are still fixed at equal distance between two consecutive centroids, or in other words as the Voronoi tesselation of the centroids), which is a necessary hypothesis in the above developments. As a consequence, result 14 is still valid in the third least restrictive case.

3.3. Weights

As in other methods to fix the parameters of RBF approximation networks, it can easily be seen that, once the centers c_j and the width factors σ_j are fixed, the problem of choosing the weight factors λ_j associated to the kernels in order to decrease the mean square error defined by 4 is purely linear, and can thus be solved for example by a pseudo-inverse or gradient descent method (see for eample in [2] or [4]).

3.4. Optimization of the parameters

It has already been mentioned that the method of splitting the search for paramaters $c(j), \sigma(j)$ and $\lambda(j)$ into three independent sets only leads to a minimization of error 4 with respect to parameters $\lambda(j)$, but not with respect to the two other sets. To avoid this drawback and to find a (local) minimum of function 4 defined with respect to the three sets of parameters, one can perform a gradient descent on function 4 simultaneously on the three sets of parameters. Using this method however suppresses the advantage of simplicity of learning which makes RBF so attractive in comparison with other neural network models. As a compromise between precision and simplicity, a gradient descent can be performed on function 4 with respect to the three sets of parameters, but taking as initial conditions the results of the splitted evaluation of parameters in the previous sections. This will thus converge to a local minimum of function 4 defined with respect to the whole sets of parameters, but one can expect that the number of iterations needed in the gradient descent process will be small since the initial conditions will be close from the convergence point. For details about this optimization procedure, see [4].

4. Conclusion

While RBF netzorks are now widely used for function approximation tasks, there is still no undisputed way to best choose the parameters used in the model. Usually, the three sets of parameters, i.e. the locations, widths and weight factors of the Gaussian kernels, are determined separately. This paper mainly addresses the problem of best choosing the width factors of the kernels, based on strong to weak assumptions on the function to approximate. A complete procedure for choosing the set of all network parameters is also given, including a global optimization of the coefficients found by splitting this set into three parts.

Acknowledgements

Michel Verleysen is Research Fellow of the Belgian National Fund for Scientific Research (FNRS). Part of this work has been funded by the ELENA ESPRIT Basic Research Action of the European Commission.

References

[1] K. Hlaváčková and R. Neruda. Radial basis functions networks. *Neural Network World*, 1:93–101, 1993.

[2] J. Moody and C. Darken. Learning with localized receptive fields. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, San Mateo, CA, 1989. Morgan Kaufmann.

[3] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[4] M. Verleysen and K. Hlavackova. An optimized RBF network for approximation of functions. In M. Verleysen, editor, *ESANN-European Symposium on Artificial Neural Networks*, pages 175–180, Brussels, Belgium, April 1994. D facto publications.