IMPLEMENTATION OF RBF TYPE NETWORKS BY SIGMOIDAL FEEDFORWARD NEURAL NETWORKS

BOGDAN M.WILAMOWSKI
University of Wyoming
RICHARD C. JAEGER
Auburn University

ABSTRACT:
It is shown that by introducing special transformations to input patterns, it is possible to separate patterns in the input space by circles, spheres, or hyperspheres. Two basic transformations are discussed. In the first transformation, the dimensionality of patterns increases by one. Since the computation of an additional input variable is rather complicated, this transformation is useful for off-line data preparation. The second transformation doubles the input pattern dimensions, but the required computation is relatively simple, so it can easily be implemented into hardware. The proposed approach is visualized with a simple two-dimensional example. It is also shown that the usually difficult problem of the separation of two spirals can be easily solved with the suggested approach.

INTRODUCTION

A single neuron is capable of separating input patterns into two categories, and this separation is linear. To separate one cluster in two dimensional space, at least three neurons in the input layer are required. To separate a cluster in three dimensional space, at least four planes (neurons) should be used. In general, to select just one region in *n*-dimensional input space, more than *n+1* neurons in the input layer should be used. If more input clusters should be selected, then the number of neurons in the first hidden layer should be properly multiplied. When radial basis functions - RBF "neurons" [Moody, 89][Hartman, 90][Girosi, 91] are used, the cluster separation problem can be significantly simplified. The RBF "neuron" is able to calculate a distance between the input pattern and the stored vector. The output signal has a maximum equal to1 when these two patterns are identical:

$$o = \exp\left| -\frac{(\mathbf{x}-\mathbf{t})^T(\mathbf{x}-\mathbf{t})}{2\sigma^2} \right| \qquad (1)$$

where $\boldsymbol{x}$ is the input pattern, $\boldsymbol{t}$ is the stored pattern, and σ is the "radius" of the cluster. Therefore, RBF "neurons" are able to separate patterns in the input space by circle, sphere and hypersphere. This feature makes the RBF network very simple and powerful in pattern recognition. Unfortunately, RBF "neurons" behave differently than biological neurons. Actual neurons cannot compute a distance between an input and the stored pattern, and neural networks with sigmoidal type activation functions require many neurons to do such clustering.

The purpose of this paper is to develop a neural network structure with typical sigmoidal neurons which are able to separate input patterns by circles, spheres or hyperspheres. This can be done by a transformation of input patterns, such that one sigmoidal type neuron is able to separate a cluster in the transformed input space. This task can be accomplished by transforming the input space on the hypersphere where each cluster can be cut out by a plane, in

a similar manner as any fragment of an apple's surface can be easily separated with a single straight cut. A similar concept of using a sphere for easy pattern separation was introduced by Kohonen [Kohonen, 87]. In this approach, input patterns are transformed into a sphere by normalizing the length of the input vectors to unity. Unfortunately, the important information about the input vector magnitude is lost and not used for pattern recognition. In order to preserve information about input patterns, the dimensionality of the input space must be increased by one, at least. As it was pointed out in the early work of Cover [Cover, 65], the nonlinear problem is more likely to be linearly separable if the size of the dimensions increase. This idea was first demonstrated by Nilson [Nilson, 65], and then implemented in the "functional link" networks by Pao [Pao, 89].

Wilensky and Manukian proposed and patented the Projection Neural Network [Wilensky, 92] where two different transformations from an *N*-dimensional input to the *N*+1 dimensional transformed input space were introduced. Both transformations use rational, square and square root functions and all dimensions have to be recalculated. Different trigonometric transformations for the same purpose were also presented [Wilamowski, 94][Ota, 94]. A slightly different approach is used in the Casasent Networks [Casasent, 92][Sarajedini, 92]. The Casasent networks increase the input dimensionally by introducing an additional variable x_{n+1} equal to the square of the magnitude of the input pattern

$$x_{n+1} = \mathbf{x}^T\mathbf{x} = x_1^2 + x_2^2 + \Lambda + x_n^2 \tag{2}$$

Sarajedini and Hecht-Nilsen presented excellent theoretical proof that a RBF network with a single hidden layer can be replaced by a Casasent network with two hidden layers [Sarajedini, 92]. In Casasent networks, the input pattern is projected on a cylindrical hyperparabola, and a cluster on the top of this cylindrical parabola can be easily separated by a linear cut of a sigmoidal neuron.

It will be shown, that by a simple increase of the problem's dimensions by one, patterns in the input space can be separated by a circle, sphere or hypersphere. The proposed approach is similar to the Casasent network, but the input space is transformed to a hypershpere, not a cylindrical hyperparabola. Furthermore, if dimensions are doubled, then the required input transformation can be realized with neurons having typical sigmoidal activation functions.

INPUT DATA TRANSFORMATION TO *N*+1 DIMENTIONAL SPACE

The input pattern transformations using trigonometric functions [Wilamowski, 94] [Ota, 94] require each input variable to be transformed. Another possible approach is to leave all input variables untouched, and introduce only one additional variable using the following transformation:

$$\begin{aligned} z_1 &= x_1 \\ z_2 &= x_2 \\ &\ldots. \\ z_n &= x_n \\ z_{n+1} &= \sqrt{r^2 - \sum_{i=1}^{n} x_i^2} \end{aligned} \tag{3}$$

where $\boldsymbol{x}$ is the n dimensional pattern in the input of the Cartesian space, $\boldsymbol{z}$ is the $n+1$ dimensional pattern transformed onto a hypersphere, and r is the radius of the hypersphere. To

avoid a negative value under the root in equation (2), it is satisfactory that radius r is larger than $\sqrt{n}\ max\{x_i\}$.

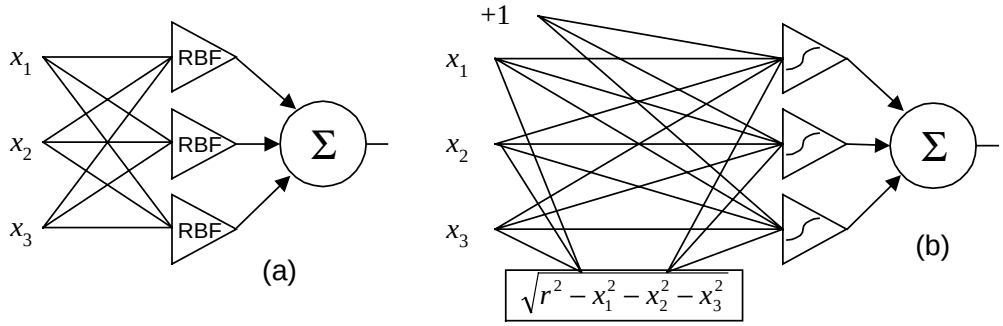

Fig. 1. Networks diagram (a) RBF network, (b) sigmoidal feedforward neural network

The patterns of the transformed z space have the same magnitudes and lie on the $n+1$ hypersphere. Each cluster can now be cut out by a single hyperplane in the $n+1$ dimensional space. The separation hyperplanes should be normal to the vectors specified by the the clusters' centers of gravity zc_k. Equations for the separation plane of the *k-th* cluster can be easily found using the point and normal vector formula:

$$\mathbf{zc}_k^T\left(\mathbf{z} - \mathbf{ze}_k\right) = 0 \qquad (4)$$

where $\mathbf{zc}_k$ is the center of gravity of the vector and $\mathbf{ze}_k$ is a point transformed from the edge of the cluster. To visualize the problem let us consider a simple two dimensional example with three clusters shown in Fig. 2. Centers of gravities $\mathbf{xc}_k$ ([5, 6] [3, -5] [-2.4, 3.8]) and points on the cluster edges $\mathbf{xe}_k$ ([3, 6] [5, -5] [-0.2, 3.8]) are also shown in this figure. Corresponding $\mathbf{zc}_k$ and $\mathbf{ze}_k$ vectors in transformed z space can be found using transformation (3).

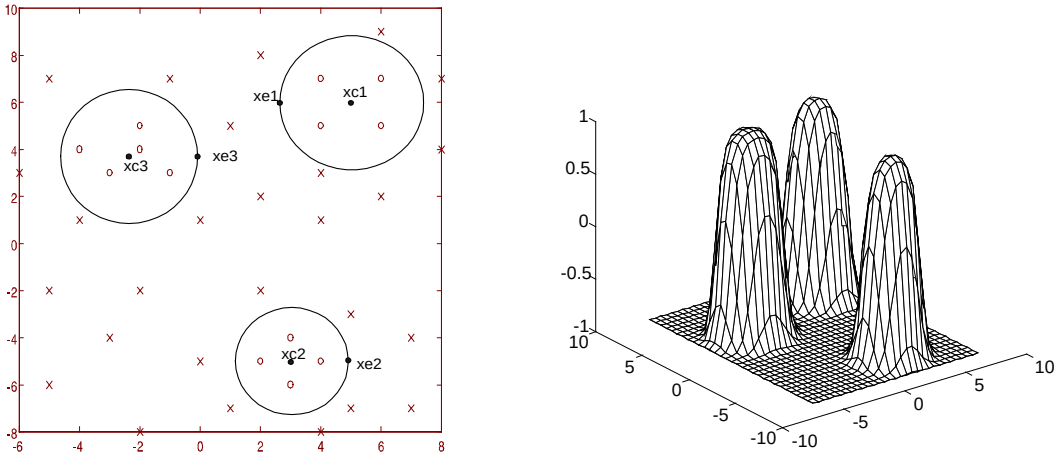

Fig. 2. Separation of three clusters (a) input space where centers of clusters and cluster edges are marked, (b) input-output mapping

Note that once the hyperplane equation is known, weights in the input layer are given by

$$\begin{aligned} w_{k,i} &= zc_{k,i} & i &= [1\,..\,n] \\ w_{k,i} &= -\mathbf{zc}_k\,\mathbf{ze}_k & i &= [n+1\,..\,2n] \end{aligned} \qquad (5)$$

The functional link network structure for separating the three clusters in Fig. 1 is shown in Fig. 2(b). The weights for the first layer, assuming the radius of hypersphere to be $r = 13$, are calculated using equations (3) and (5). Input-output mapping for this network, with neuron gains $\lambda = 1$, is shown in Fig. 2(b).

Instead of the analytical approach, the weights can also be found using the training process. The learning process can be significantly accelerated if it is assumed that the second layer performs only logical OR operations [McCulloch, 43], and only the first layer is trained. Also, an analytical solution can be treated as initial conditions for the final tuning with the error backpropagation or other training algorithm.

TRANSFORMATIONS TO *2 N* DIMENSIONS

The approach described above requires an off line computation of an additional input variable. It is also possible to simplify the required computations if transformations to higher dimensions are introduced:

$$\begin{aligned} z_i &= x_i & i &= [1 \,..\, n] \\ z_i &= \sqrt{r_j^2 - x_i^2} & i &= [n+1 \,..\, 2n] \end{aligned} \tag{6}$$

where r_j are radii of the transformation circles. For each input signal, the additional input is generated using a circle as a transformation function. In general for each component of the input vector, a different radius r_j may be used, but all values of r_j can be the same as well. Various radii should be used when components of the input vector significantly differ in magnitudes. In this case, clusters in the input space x can be separated by hyper ellipsoids.

Note that for transformation (6), it doesn't matter if the same or different radii are used, because the following equation always holds:

$$\sum_{i=1}^{n+1} z_i^2 = const \tag{7}$$

and all vectors in the transformed z space have the same magnitude and they lie on the hypersphere. Weights for the input layer can be calculated using equations (6) and (5).

The nonlinear operators required in the transformation layer

$$z_i = \sqrt{r_i^2 - x_i^2} \tag{8}$$

can also be approximated using neurons with sigmoidal type of activation functions. From the view point view of pattern separation it is not important that this is an ideal circle. As it is shown in Fig. 3, a quarter of the circle can be approximated by an inverted sigmoidal function. Instead of transformation (6), the following transformation can be used:

$$\sqrt{1 - \left|\frac{x_i}{r_i}\right|^2} \approx 1 - \frac{1}{1 + \exp\left[5\left(\frac{x_i}{r_i} - 0.866\right)\right]} \tag{9}$$

and sigmoidal type of neurons can be used for the transformation. The weights for the hidden layer can be calculated using (8) and (5).

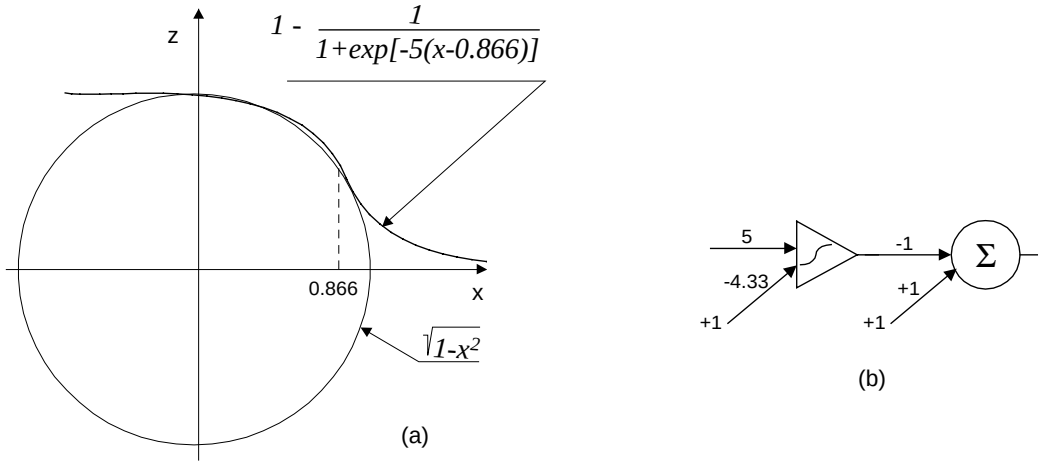

Fig 3. Function approximation (a) function comparison, (b) network to implementation- output weights and the sum operator can be incorporate into weights of the second layer.

One very difficult problem for the conventional multilayer neural network is to separate patterns laying on two neighboring spirals. With the proposed approach, the two spiral problem is relatively easy to solve. Fig. 4(a) shows the sigmoidal feedforward neural network for the two spiral problem. Fig. 4(b) is the resulting input-output mapping.

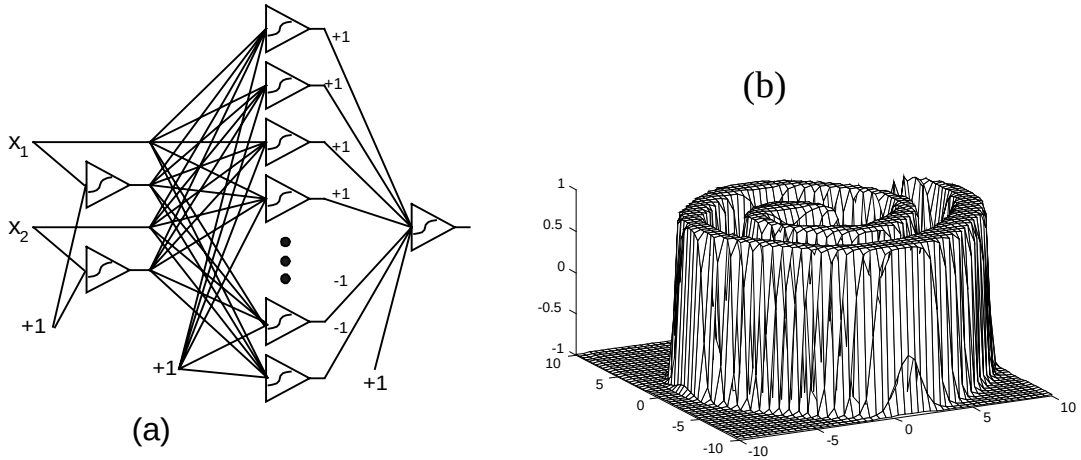

Fig. 4. Spiral problem solved with sigmoidal type neurons (a) network diagram, (b) input-output mapping.

CONCLUSION

Although the two-dimensional case was chosen for the examples, so that the input-output mapping could be easily visualized, the proposed approach will work well for multidimensional problems. For cases when input data can be transformed off line, the size of the network has to increase by one. When all processes must be implemented in hardware, it is more practical to double the dimensionality of the input patterns and use sigmoidal neurons in the transformation layer.

Weights can be found analytically or by a training process. In many practical cases, when a number of clusters and their locations are known, only one layer has to be trained. This can significantly speed up the training procedure. The key issue is to find the size and cluster

location. Unsupervised clustering methods such as SOM [Kohonen, 90], ART [Carpenter, 91] with its derivatives, and mountain clustering [Yager, 94] can be used to find the cluster locations. Unfortunately unsupervised algorithms can perform well only when all clusters are clearly defined and separated. The supervised clustering method [Wilamowski, 95] is an attractive alternative and usually leads to better results.

REFERENCES

Broomhead D. S. and D. Lowe, "Multivariable functional interpolation and adaptive networks." *Complex Systems* vol. **2**, pp. 321-355, 1988.

Carpenter G. A., S Grossberg, D. Rosen, (1991) ART 2-A: An adaptive resonance algorithm for rapid category learning and recognition, *Neural Networks*, vol. **4**, 493-504.

Casasent D. and E. Barnard, (1992) Adaptive clustering neural net for picewise nonlinear discriminant surfaces, *International Joint Conference on Neural Networks*, **I**, 423-427.

Cover T. M., "Geometrical and statistical properties of systems of linear inequalities with application to pattern recognition." *IEEE Trans. on Electronic Computers*, **EC-14**, pp. 326-334, 1965

Girosi F., T. Poggio, and B. Caprile, (1991) "Extension of a theory of networks for approximation and learning: outliners and negative examples." *Advances in Neural Information Processing Systems* 3, eds. R. P Lippmann, J. E. Moody, and D. S. Touretzky, () pp. 750-756, San Mateo, California: Morgan Kaufmann.

Hartman E., J. D. Keeler, and J. M. Kowalski, (1990) "Layered neural networks with Gaussian hidden units as universal approximations." *Neural Computation*, vol. **2**, no. 2, pp. 210-215.

Kohonen T., (1987)"Adaptive, associative, and self-organizing functions in neural computing," *Applied Optics*, vol. **26**, pp. 4910-4918.

Kohonen T., (1990)"The self-organizing map," *Proc. IEEE*, vol. **78**, no. 9.

McCulloch W. S. and W. H. Pitts, "A logical calculus of the ideas imminent in nervous activity," *Bull Math. Biophy*. vol. **5**, pp 115-133, 1943.

Moody J. and C. J. Darken, (1989) "Fast learning networks of locally-tuned processing units," *Neural Computation* vol. **1**, no. 2, pp. 281-294.

Nilson N. J., (1965) Learning Machines: Foundations of Trainable Pattern Classifiers, New York: McGraw Hill.

Ota Y. and B. Wilamowski, (1994) "Input data transformation for better pattern classification with less neurons," *Proc. of World Congress on Neural Networks*, San Diego, California, vol. **3**, pp 667-672.

Pao, Y. H. *Adaptive Pattern Recognition and Neural Networks*, Reading, Mass. Addison-Wesley Publishing Co. 1989

Sarajedini A., R. Hecht-Nielsen, (1992) The best of both worlds: Casasent networks integrate multilayer perceptrons and radial basis functions, *International Joint Conference on Neural Networks*, **III**, 905-910.

Wilamowski B. and K. Vieira, (1995) Clustering of patterns using radial base function networks, *Artificial Neural Networks in Engineering ANNIE'95*, pp. 109-115, St. Louis, Nov. 13-16.

Wilamowski B., (1994) Fast Algorithms for Neural Network Design and Training, *Proc. of the Third Intelligent Information Systems Workshop*, Wigry, Poland, pp. 443-458, June 6-10.

Wilensky G. and N. Manukian, (1992) The Projection Neural Networks, *International Joint Conference on Neural Networks*, **II**, 358-367.

Yager R. and D. Filev, (1994) Generation of fuzzy rules by mountain clustering, *Journal of Intelligent and Fuzzy System*, vol. **2**, no. 3, pp. 209-219.