ELSEVIER Neurocomputing 19 (1998) 151–165 NEUROCOMPUTING

Certainty factors versus Parzen windows as reliability measures in RBF networks

Donald K. Wedding II*, Krzysztof J. Cios

University of Toledo, Toledo, OH 43606, USA

Received 15 April 1997; accepted 23 August 1997

Abstract

A method is described for using Radial Basis Function (RBF) neural networks to generate a certainty factor reliability measure along with the network's normal output. The certainty factor approach is then compared with another technique for measuring RBF reliability, Parzen windows. Both methods are implemented into RBF networks, and the results of using each approach are compared. Advantages and disadvantages of each approach are discussed. Results indicate that certainty factors are easy to compute and good reliability measure. © 1998 Elsevier Science B.V. All rights reserved.

Keywords: RBF neural networks; Certainty factors; Parzen windows; Reliability measures; Pattern recognition

1. Introduction

A growing area of interest in the field of radial basis function (RBF) neural networks [3, 12, 14] is designing the network to generate a reliability or confidence measure along with its normal output. This is due to the fact that an RBF network gives good results for input which is similar to its training data but poor results for input that is not similar. The purpose of using confidence measures is to flag input vectors that are unfamiliar to the RBF network and discard their output as unreliable, thus increasing the overall accuracy of the output.

A method proposed by Leonard et al. [9] is to generate a confidence measure using Parzen windows. Although this method is effective in most applications,

* Corresponding author. E-mail: kcios@eng.utoledo.edu

extreme cases are shown in this paper to exist where the equations which generate the Parzen value yield results that approach zero. The extremely small values are subjected to being rouned to zero in their computer applications, which can make them useless.

This paper presents a new method of determining reliability which uses the RBF network to generate certainty factors. An RBF neural network is designed to generate both Parzen window and certainty factor values along with its normal output. It is then demonstrated that the certainty factor approach gives equally valid results as the Parzen windows for good data sets. Next it is shown that in certain cases, the Parzen window method cannot generate reliability measures whereas the certainty factor approach still gives good results.

2. Overview of RBF neural networks

The RBF neural network is composed of three layers of nodes. The first layer feeds the input data to each of the nodes in the second or hidden layer. The second layer of nodes differs greatly from other neural networks in that each node represents a data cluster which is centered at a particular point with a given radius. Each node in the second layer calculates the distance from the input vector to its own center. The determined distance is transformed via some basis function and the result is output from a node. The output from the node is multiplied by a constant or weighting value and fed into the third layer. This layer consists in its simplest case of a single node which adds the outputs of the previous layer and uses the sum to calculate an output value. A graphical representation of an RBF neural network with three data centers is shown in Fig. 1.

The more formal description of an RBF neural network follows. When the network receives a K dimensional input vector X, the network computes a scalar value using the formula

$$f_r(\bar{x}) = \lambda_0 + \sum_{i=1}^{N} \lambda_i \phi(v_i). \tag{1}$$

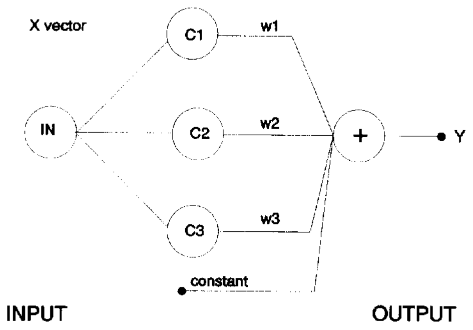


Fig. 1. Radial basis function neural network.

In Eq. (1), the value N refers to the number of nodes in the hidden layer of the RBF network. The lambdas in the formula are the weighting parameters. The vector $\boldsymbol{v}$ is of length N and represents the distance of the input vector $\boldsymbol{x}$ to each of the centers in the hidden layer. Usually, the Euclidean norm is used to calculate distance, but any other metric can also be used. The Euclidean is calculated as

$$v_i = \sqrt{\sum_{j=1}^{K} (x_j - c_{ij})^2}. \tag{2}$$

The value c_{ij} in Eq. (2) represents a cluster center for any of the given nodes in the hidden-layer, where i is the hidden-layer node number and j is the dimension. Lastly, the function $\phi(v)$ in Eq. (1) represents the basis function that the hidden-layer nodes use to transform the distance value before multiplying by the lambda values. The function chosen for this application is the Gaussian function

$$\phi(v) = \exp^{-(v^2/\sigma^2)}. \tag{3}$$

Note that in Eq. (3), σ is the radius of the cluster represented by the node.

3. RBF confidence measures

As stated above, an RBF network will give good results for input data that are similar to its training data, but will give poor results for unfamiliar input data. If the familiar and unfamiliar input data can be flagged by the network, then the user of the RBF network can disregard any output that comes from the unfamiliar data. By doing so, the overall accuracy of the network can be increased if desired.

The concept of generating a reliability measure from an RBF neural network is not a new one. Lee [8] used this principle in a handwritten digit recognition program and achieved good results. Lee used a delta measure for a confidence value which was defined to be the difference between the two output nodes with the highest values. The obvious disadvantage of this approach is that the delta can, in some instances, be large even though the actual network familiarity with the input data is low.

Leonard et al. [9] approached the problem differently. They suggested computing the reliability from the output of the RBF network's basis functions, i.e. the output from the hidden layer. They proposed adding additional nodes to the output layer of Fig. 1. This new network, shown in Fig. 2, computes both output and reliability in parallel. The user then has the choice, based upon the numeric confidence-level value, to accept or reject the output. They then examined two possible methods for calculating a reliability measure for networks using Gaussian basis functions: the maximum activation function and Parzen windows. They found the Parzen window method superior as it took into account the distribution of the training data.

Wedding and Cios [17, 18] also used the output of Gaussian basis functions to calculate a certainty factor value. The certainty factor was calculated from the input vector's proximity to the hidden layer's node centers rather than the data distribution

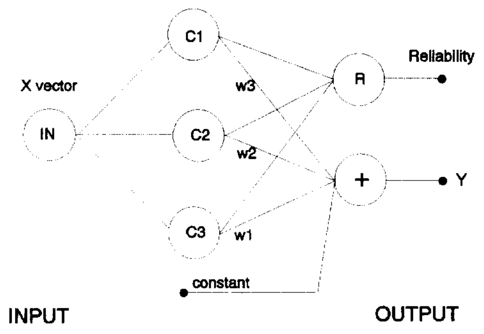


Fig. 2. RBF neural network modified to generate certainty factors along with normal output.

of the training vectors. They then used their certainty factor output to combine RBF network time-series predictions with forecasts from the Box–Jenkins method.

3.1. Parzen window reliability measures

The method of estimating the density of an unknown probability distribution function was first described by Parzen [13] and has since been used throughout the various areas of statistics. Parzen windows were eventually incorporated into the field of neural networks because they allowed analysis of complex data of which little information was available. The Parzen window was implemented specifically into RBF neural networks by Sprecht [15] who described the method, and Traven [16], who used them to estimate probability densities.

Leonard et al. [9] proposed that Parzen windows could be used to calculate a reliability measure of an RBF neural network. The reliability measure would be calculated along with the RBF decision value. The general formula for calculating the Parzen value for an input vector into the RBF network was given as

$$\rho(\bar{x}) = \frac{\sum_{i=1}^{N} \phi(v_i)\rho_i}{\sum_{i=1}^{N} \phi(v_i) + 1 - \max(\phi(v_i))} \tag{4}$$

where N is the number of nodes in the hidden layer, and $\phi(v)$ is the output from the given node in the hidden layer after applying Eqs. (2) and (3), respectively. The N values of ρ_i are calculated after the RBF network is trained using the formula

$$\rho_i = \frac{\sum_{t=1}^{T} \phi(v_i)}{T(\pi^{1/2}\sigma_i)^K} \tag{5}$$

where T is the number of vectors used to train the network, K is the dimension of vector $\boldsymbol{x}$, and σ_i is the radius of the node.

For any input vector of the RBF network, both the decision value from Eq. (1) and the ρ value of Eq. (4) must be calculated. If the value of ρ is not greater than an arbitrarily chosen threshold level, then the decision value of the RBF network is not considered reliable. Although any threshold value can be used, Leonard et al. [9] recommend that the ρ values for all T values of the training data set be calculated using Eq. (4). They suggest that a good heuristic is to use the minimum ρ value of the training data as a threshold level. When the Parzen value of an unknown input vector X is higher than that minimum, it would then be considered reliable. When it is lower, then little confidence should be placed in the network's decision value.

3.2. Certainty factor reliability measures

Certainty factors were developed by Buchanan and Shortliffe [4] for a medical diagnostic system. Certainty factors are based upon heuristic measurements of belief and disbelief of facts, and are loosely related to probability theory. To explain the concept, assume that some fact, X, is asserted to be true. If the certainty factor of this assertion is 1.0 then it is believed to be 100% true. If the certainty factor of this assertion is 0.0, then it is believed to be 0% true. Any degree within the range of 0.0–1.0 inclusive is allowed, and can all be interpreted to mean percent belief that a fact is true.

Certainty factors can be generated by RBF networks by using the $\phi(v)$ values from the basis functions in the hidden layer. Referring to Eq. (3), it is obvious that this function returns values between 1.0 and 0.0 which is also the same range as allowed for by certainty factors.

When the output $\phi(v)$ of a hidden layer node is high (near 1.0), then this indicates that a point lies near the center of a cluster and therefore the network is confident in the answer. A value of 1.0 exactly indicates that the data falls directly upon the center of a node. A value that is very small (near 0.0) will lie far outside of a cluster, so the network is not confident in the answer. If all the nodes have a small $\phi(v)$ value, then the network's certainty in the answer is small and the network is not confident in the result.

In this scheme, N values of $\phi(v)$ are generated, one for each node in the hidden layer of the RBF neural network. The question arises of how a confidence measure should be determined from all these values. There are many possible methods of combining them into a certainty factor. The obvious approach is to take the maximum $\phi(v)$ and consider it to be the certainty factor.

$$\mathrm{CF} = \max(\phi(v_i)) \tag{6}$$

This equation is the same as the maximum activation-value function proposed by Leonard et al. [9] to measure if enough training data was in the vicinity of the test vector for the neural network to make a reliable prediction. In their work, they asserted that the use of the maximum value is not an ideal measure of the sufficiency of training because it only indicates that some data are in the area of the test point, and not the amount or density of the training data in the region.

From a certainty factor standpoint, the disadvantage of Eq. (6) is that it only takes into account the maximum $\phi(v)$ value when determining the certainty factor. This may result in an instance when two or more of the hidden-layer nodes are reasonably familiar with an input data, but neither has a value close to 1.0. Therefore, the calculated certainty factor would be lower than expected which could cause good results to be discarded.

In order to factor in the certainty of other nodes, a variation of this equation is used which calculates the certainty using the recurrsive formula

$$\mathrm{CF}_i = \mathrm{CF}_{i-1} + (1 - \mathrm{CF}_{i-1})\max_i(\phi(v)) \tag{7}$$

In Eq. (7), the value of i refers to the number of nodes that will be used to determine the certainty factor of the RBF network. In general, the value of i should be set to N (the number of nodes in the hidden layer), but it can also be set to a lower value if computational time is critical to the application. The term CF_{i-1} refers to the certainty factor of the network using only $i - 1$ nodes to calculate the confidence. The value of CF_0 (the certainty when no nodes in the hidden layer are used) is defined to be 0.0. Lastly, the term $\max_i(\phi(v))$ is defined as the ith largest value of $\phi(v)$. So, $\max_1$ would mean the largest value of $\phi(v)$ and $\max_2$ would be the second largest value of $\phi(v)$, etc. The following example, which illustrates the method, was previously published by Wedding and Cios [18], and is included below for the convenience of the reader.

Example. Assume that a neural network has only four nodes in its hidden layer. For a given input vector, assume that the four $\phi(v)$ values calculated were 0.6, 0.1, 0.5, and 0.2. The certainty factors for the values of $i = 1 \dots 4$ are given in Table 1. From Table 1, it is observed that incorporating each additional hidden-layer node increases the certainty factor by a smaller and smaller amount until the network reaches its final certainty factor value. This example of diminishing returns indicates that it is not necessary to actually use $i = N$ to arrive at the optimum value of network certainty. A value of less than N will likely give an answer close to the "true" certainty. Setting the value of i too low, however, can give bad estimates. A value of $i = 1$ (which would be analogous to using Leonard et al. [9] maximum activation function) would greatly underestimate the network's certainty factor. It should be noted, however, that the most accurate results will occur when i is set to N. Approximating with values smaller than N should only be done when the time for calculating the $\max_i$ values becomes a significant factor in obtaining the network's output.

Table 1
RBF certainty factor value using i nodes of the four-node hidden layer

i	CF_{i-1}	$1 - \mathrm{CF}_{i-1}$	$\max_i(\phi(v))$	CF_i
1	0.00	1.00	0.60	0.60
2	0.60	0.40	0.50	0.80
3	0.80	0.20	0.20	0.84
4	0.84	0.16	0.10	0.86

3.3. Comparison of Parzen windows and certainty factors

Although both the Parzen window and certainty factor reliability methods have been used successfully with RBF neural networks, there are some disadvantages that should be considered when using either one in an application.

The advantage of Parzen windows over certainty factors is in applications where the data distribution is an important consideration. The certainty factor method treats all data points as being equal, so the reliability measure will be no higher for an input vector seen many times in the training data set than for an input vector seen once during training. Using Parzen windows as a reliability measure makes the more desirable assumption that the distribution of the data in the training set is representative of the actual set of input vectors.

One of the drawbacks of choosing Parzen windows over certainty factors is that it was observed [11] that the Parzen measure improves as the number of nodes in the hidden layer increases. Therefore, to achieve the optimal measure of reliability a larger number of nodes should be used than is required. This is contrary to recommendation of Weiss and Kulikowski [19], who suggested using as few nodes as possible so as to fit only data and not noise. Bishop [2] also cautioned against overfitting the data points so as to include the signal's noise as well as the signal itself. Since the certainty factor approach is a measure of how familiar each node is with the input vector, there is no requirement that a large amount of nodes be used in the hidden layer.

A more practical problem with Parzen windows occurs when the input vector is of a high dimension. From Eq. (5), it is seen that the dimension of the input vector appears as an exponent in the denominator. For high-dimensional vectors, the denominator will be large while the numerator will still be small. The resulting values calculated for ρ_i will approach zero, and although the mathematics is still valid, computers will tend to round off the values of ρ_i to zero. Applying these values into Eq. (4), the resulting reliability measure ρ will be zero for all input vectors. Therefore, the Parzen method's reliability measure for input vectors of large dimensions may cease to be meaningful. Certainty factors do not suffer from this problem since they are completely independent of the dimension of the input vector.

4. Results and discussion

An RBF neural network capable of calculating certainty factors and Parzen windows along with its normal output was implemented and tested on three data sets: the Wisconsin breast cancer database [1, 10, 11, 20], the thyroid gland database [6], and the sonar, mines versus rocks database [7].

The RBF neural network was used as a classifier to determine the classification membership of input vectors. For an input vector, the network was set to output a value in the range 0.0–1.0 for each possible classification. The output value that was closest to 1.0, was considered to be the network's classification of the vector.

For each data set, the RBF network was trained using the data at the beginning of each set and tested on the data at the end of each set. Since the purpose of this research was only to evaluate the two reliability methods and not to design an "optimal" classifier, the number of training points was deliberately kept low so that as many points as possible could be used for testing purposes. The resulting classifiers should, therefore, be viewed as acceptable for this particular application. The centers of the hidden layers were chosen from the training data set using the orthogonal least-squares (OLS) method described by Chen et al. [5]. The radii of the nodal centers were set to a single constant value for the sake of simplicity and to lessen tne training time required.

After a network was trained, a test set of data was fed into it for classification. For each vector, a certainty factor, a Parzen value, and an output classification was generated. The output was then filtered so that every vector whose certainty factor was below a user specified threshold would be discarded. The accuracy of the remaining vectors was then calculated. The certainty factor threshold ranged from 0.0 (all data points accepted) to 1.0 (virtually no data accepted) in increments of 0.1.

Next, the Parzen values of all the vectors were checked against the Parzen threshold level, which was defined as the lowest Parzen value of the training data set. After pruning the test vectors whose Parzen value was below the threshold, the accuracy was calculated. Note that this only generates one accuracy rating whereas the certainty factor method will calculate an accuracy for thresholds of 0.0–1.0 inclusive.

4.1. Wisconsin breast cancer database

This database contains data for 683 patients. Each patient's data includes nine attributes with a numeric value between 1 and 10, and the patient's classification of "malignant" or "benign". The goal was to train the RBF network to use the nine attributes to determine whether the patient's classification was malignant or benign.

In this application, the first 50 patients were used as the training data set. This number was arrived at through a trial and error approach which started at a low number and gradually increased until accurate results were achieved, since it was desired to have the largest test set possible to compare the Parzen windows and certainty factors. From the training points, 33 were selected using the OLS algorithm to be the centers points in the hidden layer. The hidden-layer nodes all had radii of 0.3 and all normalized their input vectors before calculating a distance measure. Given the small number of points in the training set, it was likely unnecessary to apply the OLS algorithm to this set, but it was included for the sake of completeness.

The network was then tested on the 633 remaining patients. The results at various certainty factor threshold levels and the Parzen window results are shown in Fig. 3. For a 0.0 certainty factor cutoff (all data is accepted), 553 of the 633 patients were correctly diagnosed as either malignant or benign. This yields an accuracy rating of 87.4%.

Using the certainty factor method, the network first filtered out any patient whose diagnosis certainty factor was lower than 0.1. After the filtering process, only 363 of the original 633 patients remained. Of these 363 patients, all were correctly diagnosed

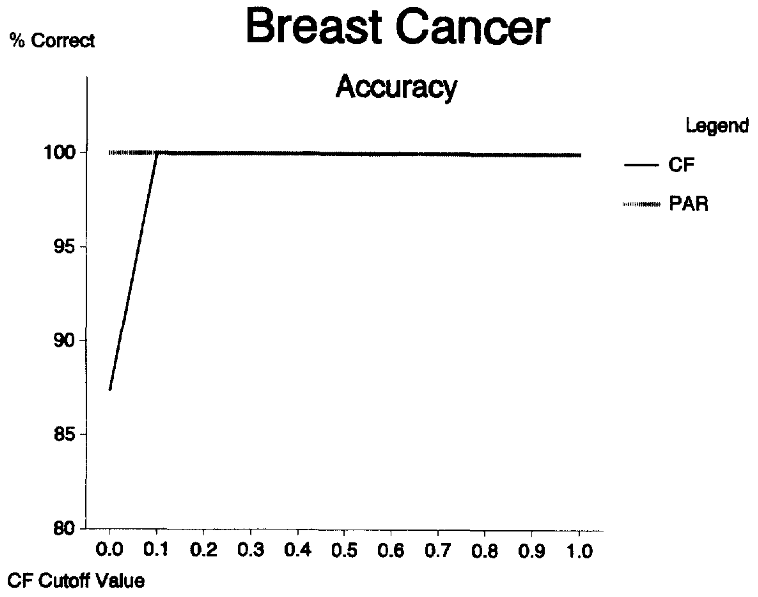


Fig. 3. Percent accuracy at varying certainty factor threshold levels.

as malignant or benign. This 100% accuracy was maintained for all certainty factor thresholds at or above 0.1 as seen in Fig. 3. Generally, a higher certainty factor cut-off is required for such accurate results. Choosing a higher level, however, would only have resulted in more data being discarded by the RBF network as unreliable as shown in Fig. 4, but would not have yielded better results. Therefore, the certainty factor threshold should be determined through trial and error on a given data set.

When the Parzen window method was employed for filtering unreliable data, the results were similar to certainty factors. The Parzen threshold level was chosen to be the smallest Parzen value of the training data, which was calculated using Eq. (4) to be 5.8853. For this threshold level, 340 of the 683 patients were diagnosed and the rest discarded as unreliable. Of these 340 patients, all were correctly diagnosed as malignant or benign. The accuracy level Parzen windows is superimposed on Fig. 3 while the percent of data meeting tolerance is shown in Fig. 4 to allow comparison with certainty factors at various cutoff levels.

From this data set, it is observed that both certainty factors and Parzen windows were successful in filtering out unreliable output to substantially increase the overall performance of the RBF neural network.

4.2. *Thyroid gland database*

The thyroid gland database [6] is a collection of 215 patients. Five laboratory tests were conducted on each patient to determine if their thyroid gland was functioning

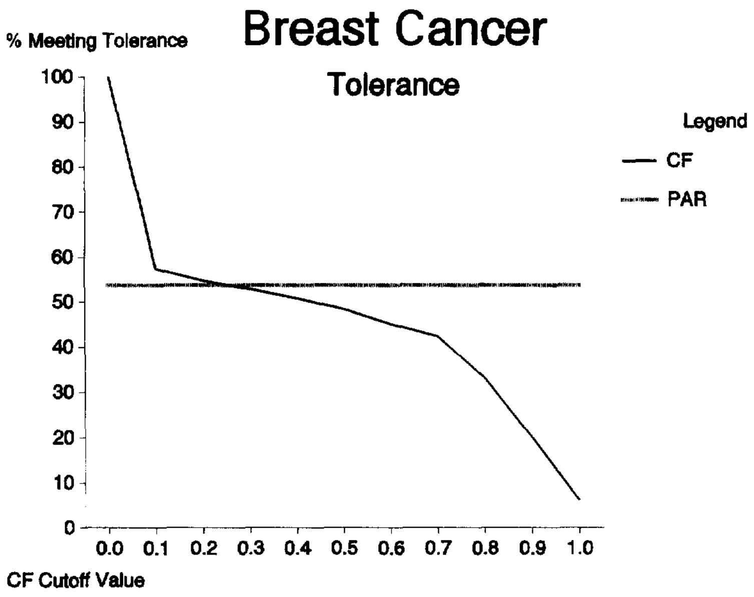


Fig. 4. Percent meeting tolerance at varying certainty factor cutoff levels.

normally, hyperactively, or hypoactively. The results were recorded as real or continuous numeric data. The database also contained the correct diagnosis of the thyroid condition which was derived from each patient's complete medical record.

The RBF network used the five test results as attributes to predict thyroid classification. The network was trained on 28 patients, and each training point was also used as a node center of the hidden layer of the neural network. The radius of each of the nodes in the hidden layers was set to a constant value of 3.0, and the input vectors of the network were not normalized.

The results at various certainty factor threshold levels are shown in Fig. 5. For a 0.0 certainty factor cutoff (all data is accepted), 158 of the 187 patients in the test set were correctly diagnosed which equated to 84.5% accuracy. The network then filtered out any patient whose diagnosis certainty factor was lower than 0.1. After the filtering process, 118 of the original 187 patients remained, and 112 of these were correctly diagnosed as having normal, hyperactive, or hypoactive thyroids. This was an accuracy rating of 94.9%. As the RBF network began filtering out patients with higher confidence levels, the accuracy rose to virtually 100%. Examining the accuracy graph of Fig. 5 and percent of data meeting tolerance graph of Fig. 6, it can be seen that this trend continues.

When the Parzen window method was employed for filtering unreliable data, the results were similar to certainty factors. The Parzen threshold level was chosen to be the smallest Parzen value of the training data. After the filtering of low Parzen values, 89 of the original 187 patients were used and 87 of them or 97.8%, were correctly

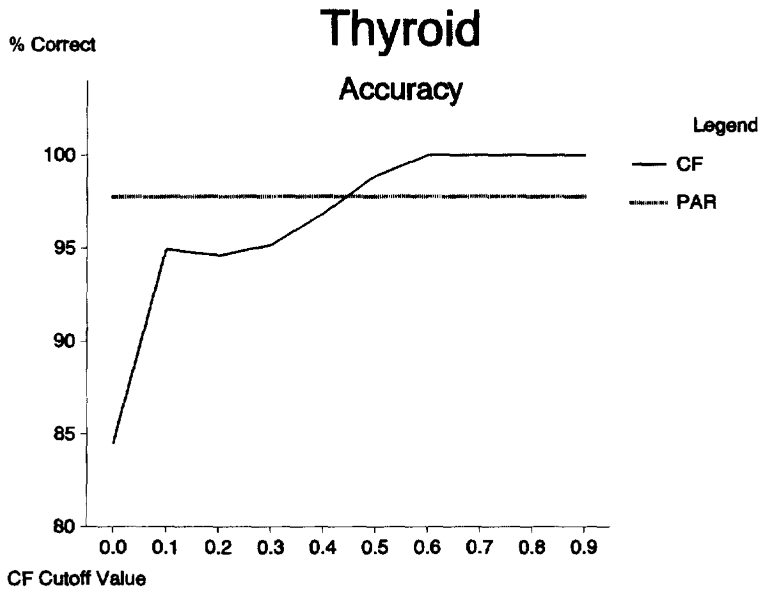


Fig. 5. Percent accuracy at varying certainty factor threshold levels.

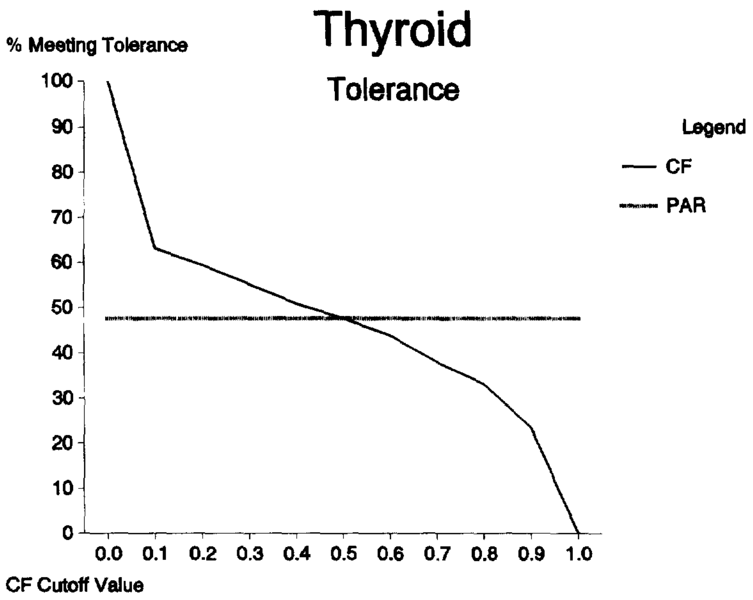


Fig. 6. Percent meeting tolerance at varying certainty factor cutoff levels.

classified. The percent accuracy is superimposed on Fig. 5 and the percent meeting tolerance is displayed in Fig. 6.

As with the Wisconsin breast cancer database, both the certainty factor and Parzen window approaches worked well in increasing the accuracy of the classifier; however, a significant difference in the Parzen windows was observed. It is seen that the radius of each of the hidden layer centers was set to 3.0, the dimension of the input vector K was 5, and the number of training vectors T was 28. When these values were input into the Parzen window equation, Eq. (5), the resulting denominator had a value of over 2 million. When this equation was used to calculate the ρ_i values, the results were very small. The largest value calculated for the $N\rho_i$ values was 0.00003. As these values were then used to determine the ρ reliability values for the training and test vectors, those results were equally small. The Parzen threshold value for this data set was set at 0.000008. Although the values were small, they still were used to successfully improve the overall accuracy of the classifier. The problem with extremely low values for ρ is that if they are too low, then the computers running the RBF network will have a tendency to round them to 0.0 which would make the Parzen window values meaningless. This situation is illustrated in the next example.

4.3. Sonar, mines versus rocks database

The sonar database [7] containes 208 entries which are numeric representations of sonar signals and their correct classification. There are 60 features in the input signals, all of which are continuous or real numbers; and one output which is the correct classification of the object that the sonar signal detected: mine or rock.

The RBF network was trained on 80 of the vectors, of which seven were selected using the OLS algorithm as node center points of the hidden layer of the network. The radii of the centers were set at a value of 1.1, and the input vectors were not normalized.

The RBF network was then tested on 128 test vectors and was able to successfully classify 73 correct entries for an accuracy rating of 57%. When filtering via certainty factors, the accuracy increased to 92% as the number of data points considered reliable dropped, as shown in Figs. 7 and 8. Again, the certainty factor method of filtering was shown to be a reliable method for improving the accuracy of the classifier.

The Parzen window approach, however, exhibited the same condition as observed in the thyroid gland database. When calculating the ρ_i values using Eq. (5), the denominator arrived at was extremely large. With the radius of 1.1, using 80 training vectors ($T = 80$) and a 60-dimensional input vector ($K = 60$), the denominator value was calculated at over 2×10^{19}. The computer used in this application rounded all ρ_i values down to 0.0. When these ρ_i values were then entered into Eq. (4) as 0.0 the resulting output was also 0.0. The significance of this is that all answers no matter how valid or invalid will always have the same Parzen reliability measure of 0.0. This means that there is no way to determine which results are reliable and which are not. Therefore, the classifier must accept all results as equally valid. The accuracy of the network cannot be improved using the Parzen windows in these cases.

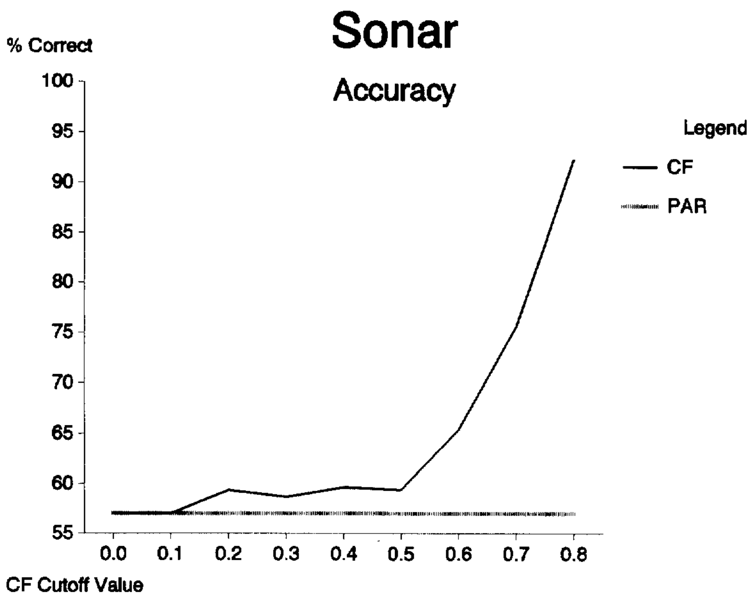


Fig. 7. Percent accuracy at varying certainty factor threshold levels.

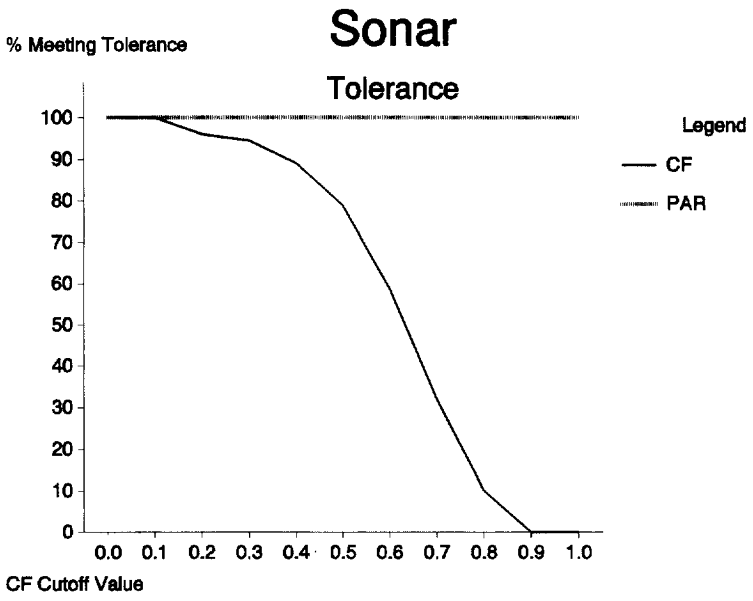


Fig. 8. Percent meeting tolerance at varying certainty factor cutoff levels.

This last example demonstrates that in extreme cases, the calculations for the Parzen window reliability measure are not feasible given the finite floating point accuracy of digital computers.

5. Conclusions

The use of reliability measures in RBF neural networks is a growing area of interest. By using only network output that has a high-reliability measure and discarding low-confidence output the overall network accuracy increases.

This paper presented a new method of determining confidence measures by using the RBF neural network to generate certainty factors. The certainty factor approach was then compared to Parzen windows as a confidence measure.

It was shown that both the certainty factor method and the Parzen window method perform equally well in circumstances when the hidden layer radii and dimensions of input vectors remain small. In these conditions, using either method results in improved RBF network performance. In the cases where the dimension of the input vector is large, the Parzen window approach is no longer capable of generating meaningful reliability measures, but the certainty factor approach still gives accurate results. Since the certainty factor approach performs well in all the cases tested while the Parzen window approach only performs well in some cases, it is concluded that the certainty factor approach is a very good method of generating confidence measures using RBF neural networks.

References

[1] K.P. Bennett, O.L. Mangasarian, Robust linear programming discrimination of two linearly inseparable sets, in: Optimization Methods and Software, vol. 1, Gordon and Breach, London, 1992, pp. 23–34.

[2] C. Bishop, Improving the generalization properties of radial basis function neural networks, Neural Comput. 3 (1991) 579–588.

[3] D.S. Broomhead, D. Lowe, multivariable functional interpolation and adaptive networks, Complex Systems 2 (1988) 321–355.

[4] B.G. Buchanan, E.H. Shortliffe, Rule-based Expert Systems, Addison-Wesley, Reading, MA, 1985.

[5] S. Chen, C.F.N. Cowan, P.M. Grant, Orthogonal least squares learning algorithm for radial basis function network, IEEE Trans. Neural Networks 2 (2) (1991) 302–309.

[6] D. Coomans, M. Broeckaert, M. Jonckheer, D.L. Massart, Comparison of multivariate discriminant techniques for clinical data – application to the thyroid functional state, Meth. Inform. Med. 22 (1983) 93–101.

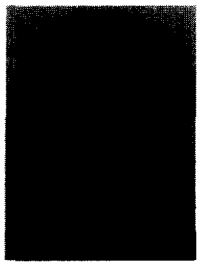
[7] R.P. Gorman, T.J. Sejnowski, Analysis of hidden units in a layered network trained to classify sonar targets, Neural Networks 1 (1988) 75–89.

[8] Y. Lee, Handwritten digit recognition using K nearest-neighbor, radial-basis function, and back-propagation neural networks, Neural Comput. 3 (3) (1991) 440–449.

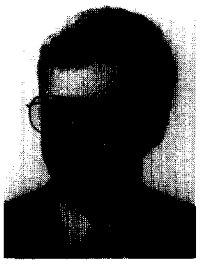
[9] J.A. Leonard, M.A. Kramer, L.H. Unger, A neural network architecture that computes its own reliability, Comput. Chem. Eng. 16 (9) (1992) 819–835.

[10] O.L. Mangasarian, R. Setiono, W.L. Wolberg, Pattern recognition via linear programming: theory and application to medical diagnosis, in: Thomas F. Coleman, Yuying Li (Eds.), Large-scale Numerical Optimization, SIAM Publications, Philadelphia, 1990, pp. 22–30.

[11] O.L. Mangasarian, W.H. Wolberg, Cancer diagnosis via linear programming, SIAM News 23 (5) (1990) 1 & 18.
[12] J. Moody, C.J. Darken, Fast learning in networks of locally-tuned processing units, Neural Comput. 1 (1989) 281–294.
[13] E. Parzen, On Estimation of a probability density function and mode Ann, Math. Statist. 33 (1962) 1065–1076.
[14] T. Poggio, F. Girosi, Networks for approximation and learning, Proc. IEEE 78 (1990) 1481–1497.
[15] D.F. Specht, Probabilistic neural networks, Neural Networks 3 (1990) 109–118.
[16] H.G.C. Traven, A neural network approach to statistical pattern classification by 'semiparametric' estimation of probability density functions, IEEE Trans. Neural Networks 3 (1991) 366–377.
[17] D.K. Wedding, Improved accuracy of RBF neural networks through the implementation of certainty factors, Ph.D. Dissertation, University of Toledo, Toledo, Ohio, 1995.
[18] D.K. Wedding, K.J. Cios, Time series forecasting by combining RBF neural networks and the box–Jenkins model, Neurocomput. 10 (2) (1996) 149–168.
[19] S.M. Weiss, C.A. Kulikowsk, Computer Systems that Learn, Morgan Kaufmann, San Mateo, CA, 1991.
[20] W.H. Wolberg, O.L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, Proc. Nat. Acad. Sci. USA, vol. 87, December 1990, pp. 9193–9196.



Donald K. Wedding received his B.S. in Electrical Engineering in 1987 and his M.S. in Engineering Science in 1988 from the University of Toledo. He then worked as a software engineer in the defense industry until 1992 when he returned to graduate school and completed his Ph.D. in Engineering Systems from the University of Toledo in 1995. Dr. Wedding has also earned an M.S. in Management Systems with an emphasis in finance in 1991. Dr. Wedding's conducts research in artificial intelligence, expert systems, neural networks, and data mining. Dr. Wedding applies his research to the fields of financial forcasting, financial analysis, legal search engines, medical systems, and photograph imaging. Dr. Wedding is presently employed by a medical systems company in Cleveland, Ohio. Dr. Wedding is also working with Kathryn Barnes of the Visual Echos company who is one of the premier artistic photographers in the United States.



Krzysztof J. Cios received an M.S. degree in electrical engineering and a Ph.D. in computer science, both from AGH Technical University, Krakow, and an MBA degree from University of Toledo. He is a Professor of Bioengineering and Professor of Electrical Engineering and Computer Science, Department of Bioengineering, University of Toledo, Toledo, OH, U.S.A. His research interests are in the area of neuroengineering, intelligent systems, and data mining. His research was funded by NASA, National Science Foundation, NATO, and the American Heart Association. He has published extensively in journals, conference proceedings, and wrote several book chapters. His paper on inductive machine learning received The Norbert Wiener 1997 Outstanding Paper Award from *Kybernetes*. Dr. Cios consults for domestic and foreign companies and gives tutorials in his areas of expertise. He is a senior member of the IEEE, and serves on several editorial boards.