

Resource-Allocating Probabilistic Neuro-Fuzzy Network

Ye. Bodyanskiy

Ye. Gorshkov

V. Kolodyazhnyi

Control Systems Research Laboratory,
Kharkiv National University of Radioelectronics,
14, Lenin Av., Kharkiv, 61166, Ukraine
E-mail: {Bodyanskiy, Yevgen.Gorshkov, Kolodyazhnyi}@ieee.org

Abstract

In this paper, an architecture of a resource-allocating learning probabilistic neural network is considered. Construction and learning algorithms are proposed. The advantages of this network lie in the possibility of classification of data with substantially overlapping clusters. The construction algorithm significantly reduces the size of the network and tuning of the activation function parameters improves the accuracy of classification. Simulation results confirm the efficiency of the proposed approach in the data classification problems.

1 Introduction

Clustering and data classification are key problems of data mining, and solving these problems is important for effective knowledge accumulation on the basis of observational analysis. One of the popular techniques of data classification are the probabilistic neural networks (PNN), proposed by D.F. Specht [11, 12]. These networks are effectively applied to various problems of classification, diagnosis, pattern recognition, etc. [3]. However, the conventional PNN solves the classification problem according to the traditional “crisp” approach, when every observation is assigned to only one cluster. But in practice the problems with overlapping clusters are common, when the classified observation may belong to several classes simultaneously with certain degrees of membership. To solve such problems, the methods based on the fuzzy approach to clustering, classification, and pattern recognition are used [2].

The neuro-fuzzy synergism [6] extends the range of the problems solved with the classifier, combining the advantages of fuzzy data analysis with the capabilities of learning from data.

In this paper, we make an attempt to modify the conventional PNN by adding the properties of fuzzy classification, learning of the activation function parameters, and constructing the network with a resource-allocation method [10, 1].

2 Network Architecture

In the general case, the probabilistic neural networks which solve the problem of Bayesian classification [4] via the recovery of unknown probability distributions by means of the Parzen kernels [4, 9], belong to the feedforward architectures and are closely related to the radial-basis function networks [7] and generalized regression neural networks [13].

The fuzzy probabilistic neural network with proposed in the paper is a three-layer structure shown in fig. 1.

The source information for the synthesis of networks is the training set of patterns, formed as a set of n -dimensional vectors $x(1), \dots, x(j), \dots, x(M)$, $x(j) = (x_1(j), \dots, x_i(j), \dots, x_n(j))^T$. It is assumed that the crisp classification (the membership to one of m clusters) is known for each of the training patterns $x(j)$, and the representatives of all the possible clusters must be present in the dataset. That is, if N_l observations from the training set belong to the l -th cluster, then

$$N = \sum_{l=1}^m N_l.$$

The input layer (number 0) receives an n -dimensional pattern vector $x(k)$, $k > M$ with unknown classifica-

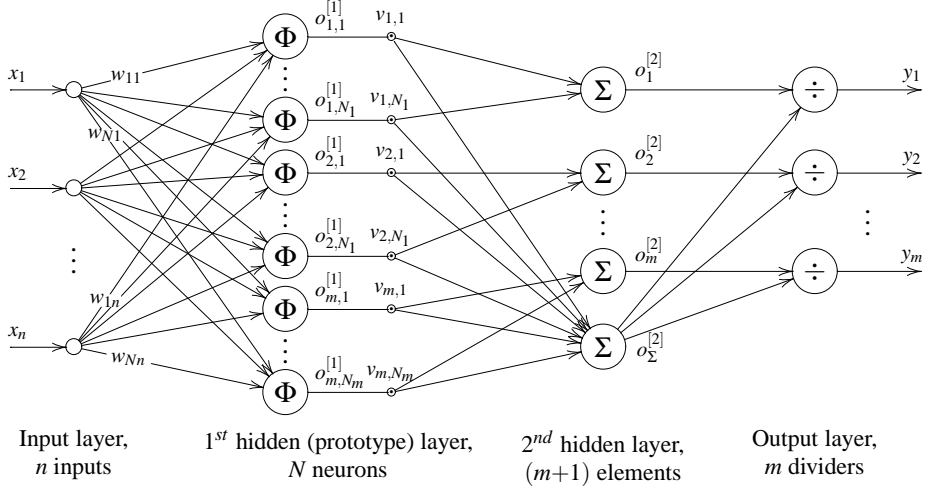


Figure 1: Probabilistic neuro-fuzzy network

tion. It is transferred to the hidden layers for further processing.

The first hidden layer (the prototype layer) contains N neurons with the bell-shaped (usually Gaussian, as in our case) activation functions, and their synaptic weights are determined by the components of the training patterns, i.e.

$$w_{ji} = x_i(j), \quad i = 1, 2, \dots, n, 1 \leq j \leq M,$$

or in the vector form

$$w_j = x(j) = (x_1(j), \dots, x_i(j), \dots, x_n(j))^T.$$

For the convenience of notation, all the neurons in the prototype layer are divided into m groups with N_l nodes in each, corresponding to its cluster. The vector of weights of the p -th neuron in the l -th group will be denoted as $w_{l,p} = w_{l+p-1}$, $l = 1, 2, \dots, m$, $p = 1, 2, \dots, N_l$.

It is obvious that the training of the synaptic weights of the prototype layer in this case is reduced to their one-time setting, that is very simple.

When the vector $x(k)$ is fed to the 0-th layer, the neurons of the first layer produce the signals

$$o_{l,p}^{[1]}(k) = \exp\left(-\frac{\|x(k) - w_{l,p}\|^2}{2\sigma_l^2}\right), \quad (1)$$

for $l = 1, \dots, m$, $p = 1, \dots, N_l$, $k > M$, where $w_{l,p}$ is the center of the activation function and σ_l is its receptive field radius parameter.

All the input vectors are pre-normalized so that $\|x\|=1$ and $\|w_{l,p}\|=1$. Since it holds $-1 \leq x^T(k)w_{l,p} \leq 1$, the outputs of the neurons of the first layer can vary only within the interval $\exp(-2\sigma_l^{-2}) \leq o_{l,p}^{[1]} \leq 1$.

At the level of the first layer, it is also possible to determine the diameters of the clusters formed by the training data as $0 \leq D_l = \max \|w_{l,p} - w_{l,q}\|^2 \leq 2$, and to roughly estimate how much they overlap.

The second hidden layer is formed by $m+1$ elementary summing units, with first m of them receiving the outputs of the prototype layer so that

$$o_l^{[2]}(k) = \sum_{p=1}^{N_l} v_{l,p} o_{l,p}^{[1]}(k), \quad (2)$$

where $v_{l,p} \geq 1$ is a second hidden layer synaptic weight used to determine the cluster shape (data distribution) more precisely. The output of the $(m+1)$ -th summing unit calculates the total sum

$$o_{\Sigma}^{[2]}(k) = \sum_{l=1}^m \sum_{p=1}^{N_l} v_{l,p} o_{l,p}^{[1]}(k). \quad (3)$$

It can be readily seen that the sums (2) are Parzen approximations [4, 9] of the unknown data distributions in the clusters.

Finally, the output normalization layer formed by m dividers calculates the vector of degrees of membership $y(k) = (y_1(k), \dots, y_l(k), \dots, y_m(k))^T$ of the pro-

cessed observation as

$$0 \leq y_l(k) = \frac{o_l^{[2]}(k)}{o_\Sigma^{[2]}(k)} \leq 1, \quad \sum_{l=1}^m y_l(k) = 1.$$

It is easy to see that the described network is a combination of the probabilistic and generalized regression neural networks, and is capable of data classification on the basis of fuzzy decision on the membership of a particular observation to a certain class.

It is also evident that the first two layers of this network are essentially a radial basis function architecture with fixed synaptic weights $w_{lj}^{RBF} \equiv 1$ and centers $c_j^{RBF} \equiv w_j$.

Undetermined are only the cluster width parameters σ_l^2 , which significantly affect the classification accuracy and are usually chosen empirically [5], and the number of neurons in the prototype layer (the size of the network). Note that the weights $v_{l,p}$ must be initially set to $v_{l,p} = 1$.

Therefore, it is advisable to introduce a learning algorithm to adjust the receptive field parameters σ_l^2 for more accurate approximation of the training data $x(u)$, $u = 1, \dots, M$ using their crisp classification $d(u)$, and to use resource-allocating building method to form the prototype layer with the optimal size.

3 Network Construction

The following network construction method is proposed. When for the given learning sample $x(u)$ fed to the current network all the neuron's outputs of the same class $l = \arg \max d(u)$ as $x(u)$ are below the given threshold value θ then a new neuron with center $w_{l,N_l+1} = x(u)$ of the activation function and synaptic weight $v_{l,N_l+1} = 1$ of the 2^{nd} hidden layer is added to the network so that $N_l := N_l + 1$, $N := N + 1$.

Otherwise, for the neuron q of the corresponding class l with the maximum output value $q = \arg \max_{p=1, \dots, N_l} o_{l,p}^{[1]}(u)$, the 2^{nd} hidden layer synaptic weight is incremented as $v_{l,q} := v_{l,q} + o_{l,p}^{[1]}(u)$. Then one learning step is performed on the sample $x(u)$.

Incrementing the values $v_{l,p}$ can be stopped when enough learning samples have been processed, i.e. when the data distribution in the clusters could be approximated appropriately well using kernel-regression methods [4].

4 Learning Algorithm

Let us introduce a one-step learning criterion (error function)

$$E(u) = \sum_{h=1}^m E_h(u) = \frac{1}{2} \|e(u)\|^2,$$

where $e(u) = (e_1(u), \dots, e_h(u), \dots, e_m(u))^T$, $E_h(u) = \frac{1}{2} e_h^2(u) = \frac{1}{2} (d_h(u) - y_h(u))^2$ and $d_h(u)$ is the training signal equal to 1 if the vector $x(u)$, $u = 1, \dots, M$ belongs to the h -th cluster, and 0 otherwise.

The derivatives of the error function with respect to the tunable parameters will be

$$\frac{\partial E_h(u)}{\partial \sigma_l^{-2}} = e_h(u) \frac{y_h(u) - \delta_{hl}}{o_\Sigma^{[2]}(u)} \sum_{p=1}^{N_l} v_{l,p} \frac{\partial o_{l,p}^{[1]}(u)}{\partial \sigma_l^{-2}},$$

where $h = 1, 2, \dots, m$ and $\delta_{hl} = \begin{cases} 1, & \text{if } h = l, \\ 0, & \text{otherwise.} \end{cases}$

It can be readily seen that

$$\frac{\partial E(u)}{\partial \sigma_l^{-2}} = \sum_{h=1}^m e_h(u) \frac{y_h(u) - \delta_{hl}}{o_\Sigma^{[2]}(u)} \sum_{p=1}^{N_l} v_{l,p} \frac{\partial o_{l,p}^{[1]}(u)}{\partial \sigma_l^{-2}},$$

and for the Gaussian activation function

$$\frac{\partial o_{l,p}^{[1]}(u)}{\partial \sigma_l^{-2}} = -\frac{1}{2} \|x(u) - w_{l,p}\|^2 \exp\left(-\frac{\|x(u) - w_{l,p}\|^2}{2\sigma_l^2}\right).$$

Minimizing $E(u)$ with the standard gradient-based procedure we can finally write the learning algorithm for the probabilistic network with fuzzy inference

$$\sigma_l^{-2}(u+1) = \sigma_l^{-2}(u) + \eta(u) \sum_{h=1}^m e_h(u) \frac{y_h(u) - \delta_{hl}}{o_\Sigma^{[2]}(u)} \cdot \sum_{p=1}^{N_l} v_{l,p} \|x(u) - w_{l,p}\|^2 \exp\left(-\frac{\|x(u) - w_{l,p}\|^2}{2\sigma_l^2}\right),$$

where $\eta(u)$ is the scalar learning rate parameter.

When the learning is finished, the values $\sigma_l^{-2}(M+1)$, $l = 1, 2, \dots, m$ are used as estimates of the activation function parameters (1) in the classification of objects $x(k)$, $k > M$ with unknown membership.

5 Experiments

Several well-known datasets were chosen to test the performance of the proposed neural network and learning algorithm: the Iris data, the Wisconsin breast cancer (WBC) data, and the Thyroid disease data from the UCI repository [8].

We have compared the proposed classifier with the original PNN. Our test is intended to examine the learning capabilities of the network and its classification performance.

The receptive field parameter of the PNN was chosen empirically to minimize the classification error and varies in $\sigma = 0.1 \div 0.5$ for different datasets. For the proposed network, the initial value $\sigma_l(0) = 0.5$ was chosen for all datasets. The following parameters values were set: $\eta = 80$, $\theta = 0.04$ for Iris, $\eta = 0.7$, $\theta = 0.7$ for WBC and $\eta = 0.003$, $\theta = 0.6$ for Thyroid.

Since the one-step error function is used, the training procedure can be executed in the on-line mode. But for the small training data sets (Iris and WBC) it is necessary to train the network for several epochs on the same data. In our experiments, $20 \div 25$ epochs were enough for the network to learn.

The tests on all datasets were performed 100 times each and the mean values were obtained. Mean error values for compared classifiers are given in Table 1 (mean error rate on the testing data / mean network size N).

	PNN	FPNN
Iris	3.2% / 100	4.1% / 21
WBC	3.4% / 200	2.9% / 20
Thyroid	7.3% / 3772	7.3% / 22

Table 1: Test results

It should be noted here that the classification accuracy of the proposed network significantly increases after the learning performed. However, there is still a question of determining the parameters η and θ of the learning algorithm to obtain best results. To solve this problem an optimal learning algorithm could be applied.

From the given results it can be seen that the proposed PNN provides enough classification performance, using considerably less computational resources since the size of the PNN is more than 5 times bigger.

6 Conclusion

Simulation results confirm high performance of the proposed resource-allocating learning probabilistic neural network with fuzzy output, which provides good classification using less computational resources than the conventional PNN. It is simple and has high rate of convergence of the learning procedure.

References

- [1] M. Berthold and J. Diamond. Boosting the Performance of RBF Networks with Dynamic Decay Adjustment. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 521–528, Cambridge, MA, 1995. MIT Press.
- [2] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, N.Y., 1981.
- [3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.
- [4] K. Fukunaga. *Intruduction to Statistical Pattern Recognition*. Academic Press, New York and London, 1972.
- [5] S. Haykin. *Neural Networks. A Comprehensive Foundation*. Prentice Hall, Inc., Upper Saddle River, N.Y., 1998.
- [6] J.-S. R. Jang, C.-T. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing — Computational Approach to Learning and Machine Intelligence*. Prentice Hall, Upper Saddle River, 1997.
- [7] J. Moody and C. J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computing*, 1:281–299, 1989.
- [8] P. M. Murphy and D. W. Aha. *UCI Repository of machine learning databases*. University of California, Department of Information and Computer Science, CA, 1994.
- [9] E. Parzen. On the estimation of a probability density function and the mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.
- [10] J. Platt. A resource-allocating network for function interpolation. *Neural Computation*, 3:213–255, 1991.
- [11] D. F. Specht. Probabilistic neural networks. *Neural Networks*, 3:109–118, 1990.
- [12] D. F. Specht. Probabilistic neural networks and polynomial adaline as complementary techniques for classification. *IEEE Trans. on Neural Networks*, 1(1):111–121, 1990.
- [13] D. F. Specht. General regression neural networks. *IEEE Trans. on Neural Networks*, 2:568–576, 1991.