A New Method for Constructing Radial Basis Function Neural Networks

Jinyan Sun Xizhao Wang

Machine Learning Center, Faculty of Mathematics and Computer Science, Hebei University, Baoding 071002, P.R. China

Abstract

Ignoring the samples far away from the training samples, our study team gives a new norm-based derivative process of localized generalization error boundary. Enlightened by the above research, this paper proposes a new method to construct radial basis function neural networks, which minimizes the sum of training error and stochastic sensitivity. Experimental results show that the new method can lead to simple and better network architecture.

Keywords: Radial basis function neural network, Norm, Training error; Sensitivity

1. Introduction

In pattern classification problems, generalization capability is a criterion of appraising classifier and we always hope a trained classifier can recognize unseen samples correctly. The samples, which are located far away the hidden neuron centers, have insignificant effect in the learning of hidden neuron centers [2]-[4]. Wing W.Y.NG, et al present localized generalization error model and deduce an upper boundary of localized generalization error [1]. They ignore the samples far away from the training samples and compute generalization error within a neighborhood of training samples based on mean square error as follows:

$$R_{SM} = \int_{S_Q} \left(f_\theta(\mathrm{x}) - F(\mathrm{x}) \right)^2 p(\mathrm{x})\, d\mathrm{x}$$

where x is input vector in the neighborhood S_Q, which is the union of all neighborhoods of training samples.

To further simplify the formula of localized generalization error model (LGEM) and get a smaller upper bound of localized generalization error (LGE), our study team renewedly deduces the LGE based on norm to compute the difference between the trained output function $f_\theta(\mathrm{x})$ and the target function $F(\mathrm{x})$, i.e. $R_{gen} = \int_{S_Q} \| f_\theta(\mathrm{x}) - F(\mathrm{x}) \| p(\mathrm{x})\, d\mathrm{x}$, where $p(\mathrm{x})$ is the unknown probability density function of the input x.

During research on LGEM, we find that the upper bound of LGE can be confirmed by training error and sensitivity. Training error can reflect how well the classifier study from training samples. Neural network stochastic sensitivity is useful to assess a trained neural network [5]. In [6], the experiments show that the sensitivity measure would be correlated to the testing error of unseen samples.

In this paper, we give a criterion function assessing a trained Radial Basis Function Neural Network (RBFNN) classifier, which is the sum of training error and stochastic sensitivity. And then, this paper minimizes the criterion function, i.e. the sum of training error and sensitivity (MTAS), to ascertain the appropriate RBFNN architecture. The new method MTAS considers the generalization capability of RBFNN and it is unnecessary to choose the number of hidden neuron centers in advance.

2. The norm-based localized generalization error model

Given a training dataset D containing N training samples, $D = \{(\mathrm{x}_i, \mathrm{y}_i)\}_{i=1}^N$, we compute the generalization error of unseen samples located in a neighborhood S_Q of training samples based on norm as follows:

$$R_{gen} = \int_{S_Q} \| f_\theta(\mathrm{x}) - F(\mathrm{x}) \| p(\mathrm{x})\, d\mathrm{x}$$

where

$$S_Q = \bigcup_{i=1}^{N} S_Q(\mathrm{x}_i)$$
$$= \bigcup_{i=1}^{N} \left\{ \mathrm{x} \mid \mathrm{x} = \mathrm{x}_i + \Delta\mathrm{x}; |\Delta x_l| \le Q \;\; \forall l = 1, \ldots, n \right\}$$

, Q is a given non-negative real number and n denotes the number of input features.

The following is the derivative process of the norm-based localized generalization error model (NLGEM) boundary.

$$R_{gen}=\int_{S_Q}\left\|f_\theta(\mathrm{x})-F(\mathrm{x})\right\|p(\mathrm{x})d\mathrm{x}$$
$$\le\sum_{i=1}^{N}\int_{S_Q(\mathrm{x}_i)}\left\|f_\theta(\mathrm{x})-F(\mathrm{x})\right\|p(\mathrm{x})d\mathrm{x}$$
$$=\sum_{i=1}^{N}\int_{S_Q(\mathrm{x}_i)}p(\mathrm{x})d\mathrm{x}*\int_{S_Q(\mathrm{x}_i)}\left\|f_\theta(\mathrm{x})-F(\mathrm{x})\right\|\frac{p(\mathrm{x})}{\int_{S_Q(\mathrm{x}_i)}p(\mathrm{x})d\mathrm{x}}d\mathrm{x}$$

Here, we define

$$C_i=\int_{S_Q(\mathrm{x}_i)}p(\mathrm{x})d\mathrm{x}=p(\mathrm{x}_i^{'})\Delta S_Q(\mathrm{x}_i),\ \mathrm{x}_i^{'}\in S_Q(\mathrm{x}_i)$$

$$p_{S_Q(\mathrm{x}_i)}(\mathrm{x})=\frac{p(\mathrm{x})}{C_i}$$

Because of $\int_{S_Q(\mathrm{x}_i)}p_{S_Q(\mathrm{x}_i)}(\mathrm{x})d\mathrm{x}=1$, $p_{S_Q(\mathrm{x}_i)}(\mathrm{x})$ can be considered as the probability density function of x in $S_Q(\mathrm{x}_i)$.

$$=\sum_{i=1}^{N}C_i*\int_{S_Q(\mathrm{x}_i)}\left\|f_\theta(\mathrm{x})-F(\mathrm{x})\right\|p_{S_Q(\mathrm{x}_i)}(\mathrm{x})d\mathrm{x}$$

Here, we adjust $\Delta S_Q(\mathrm{x}_i)$ to keep the relation, $C_i\le\frac{\xi}{N}(\xi\ge1)$, ξ is a control parameter of $\Delta S_Q(\mathrm{x}_i)$.

Then, we have the following derivation:

$$\le\frac{\xi}{N}\sum_{i=1}^{N}\int_{S_Q(\mathrm{x}_i)}\left\|f_\theta(\mathrm{x})-f_\theta(\mathrm{x}_i)\right\|p_{S_Q(\mathrm{x}_i)}(\mathrm{x})d\mathrm{x}$$
$$+\frac{\xi}{N}\sum_{i=1}^{N}\left\|f_\theta(\mathrm{x}_i)-F(\mathrm{x}_i)\right\|$$
$$+\frac{\xi}{N}\sum_{i=1}^{N}\int_{S_Q(\mathrm{x}_i)}\left\|F(\mathrm{x}_i)-F(\mathrm{x})\right\|p_{S_Q(\mathrm{x}_i)}(\mathrm{x})d\mathrm{x}$$

During adjust $\Delta S_Q(\mathrm{x}_i)$, i.e. Q, we try to ensure that the samples in the same Q neighborhood belong to the same class, i.e. they have the same class label. So, we have

$$\int_{S_Q(\mathrm{x}_i)}\left\|F(\mathrm{x}_i)-F(\mathrm{x})\right\|p_{S_Q(\mathrm{x}_i)}(\mathrm{x})d\mathrm{x}=0$$

Then

$$=\frac{\xi}{N}\sum_{i=1}^{N}\left\|f_\theta(\mathrm{x}_i)-F(\mathrm{x}_i)\right\|$$
$$+\frac{\xi}{N}\sum_{i=1}^{N}\int_{S_Q(\mathrm{x}_i)}\left\|f_\theta(\mathrm{x})-f_\theta(\mathrm{x}_i)\right\|p_{S_Q(\mathrm{x}_i)}(\mathrm{x})d\mathrm{x}$$
$$=R_{gen}^{*}\qquad(1)$$

Remark 1: Q can be considered as the maximum value of input perturbation. The perturbed input vectors in every neighborhood would be the unseen samples outside the training samples.

Remark 2: We adjust Q to make the samples in a neighborhood belong to the same class. The target output difference between training samples and their perturbed samples equals to zero. So, the upper boundary of LGE is predigested.

Remark 3: For a given Q, R_{gen}^{*} refers to unseen samples, which locate within the neighborhood of training samples and are similar to the training samples. R_{gen}^{*} can denote the generalization capability of a trained classifier according to its value.

3. Further derivation about boundary of LGE

Because ξ exists in the two components, for a appropriate Q, we can ignore ξ and pay attention to the following formula:

$$R_{TS}=\frac{1}{N}\sum_{i=1}^{N}\left\|f_\theta(\mathrm{x}_i)-F(\mathrm{x}_i)\right\|$$
$$+\frac{1}{N}\sum_{i=1}^{N}\int_{S_Q(\mathrm{x}_i)}\left\|f_\theta(\mathrm{x})-f_\theta(\mathrm{x}_i)\right\|p_{S_Q(\mathrm{x}_i)}(\mathrm{x})d\mathrm{x}$$
$$=R_{emp}+E_{S_Q}\left(\left\|\Delta y\right\|\right)$$

where the first component is training error and the second component is stochastic sensitivity, which measures the value of the network output perturbation with respect to input perturbations.

Here, we consider a $L-class$ classification problem and the derivations are as follows:

If we adopt 1-norm, we have the following derivation:

$$R_{TS}=\frac{1}{N}\sum_{i=1}^{N}\left\|f_\theta(\mathrm{x}_i)-F(\mathrm{x}_i)\right\|_1+E_{S_Q}(\left\|\Delta y\right\|_1)$$
$$=\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{L}\left|f_{\theta k}(\mathrm{x}_i)-F_k(\mathrm{x}_i)\right|+\sum_{k=1}^{L}E_{S_Q}(\left|\Delta y_k\right|)\qquad(2)$$

If we adopt 2-norm, we have another derivation:

$$R_{TS}=\frac{1}{N}\sum_{i=1}^{N}\left\|f_\theta(\mathrm{x}_i)-F(\mathrm{x}_i)\right\|_2+E_{S_Q}(\left\|\Delta y\right\|_2)$$
$$=\frac{1}{N}\sum_{i=1}^{N}\sqrt{\sum_{k=1}^{L}\left(f_{\theta k}(\mathrm{x}_i)-F_k(\mathrm{x}_i)\right)^2}+E_{S_Q}\left(\sqrt{\sum_{k=1}^{L}(\Delta y_k)^2}\right)\qquad(3)$$

Moreover, different norm can be used in the derivative process synchronously. For example, the first component adopts 1-norm while the second adopts 2-norm. Other norm can be adopted in the above formula, too.

4. The method for constructing RBFNN

4.1. Criterion function

Enlightened by the above research and the results in (1), we know the boundary of LGE is mainly affected by the two components, training error and stochastic sensitivity, whatever norm is adopted as (2) or (3). In this paper, we use mean square error to compute the training error and present a criterion function assessing a trained classifier. For all the samples within the S_Q, we have the following equation,

$$\begin{aligned} TS &= R_{emp} + E_{S_Q}\left((\Delta y)^2\right) \\ &= \frac{1}{N}\sum_{\mathrm{i}=1}^{N}\left(f_\theta(\mathrm{x}_i) - F(\mathrm{x}_i)\right)^2 + E_{S_Q}\left((\Delta y)^2\right) \qquad (4) \\ &= \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{L}\left(f_{\theta k}(\mathrm{x}_i) - F_k(\mathrm{x}_i)\right)^2 + \sum_{k=1}^{L} E_{S_Q}\left((\Delta y_k)^2\right) \end{aligned}$$

In equation (4), the first component is the mean square error of training samples and the second component is stochastic sensitivity, which is the expectation of square of output perturbations with respect to input perturbations. Then the criterion function will be applied to ascertain the appropriate number of hidden neuron centers and construct RBFNN.

4.2. The RBFNN stochastic sensitivity

The k-th output of RBFNN with multiple outputs could be described as:

$$\begin{aligned} f_k(\mathrm{x}) &= \sum_{j=1}^{M} w_{kj}\exp\left[\frac{\sum_{l=1}^{n}\left(x_l - u_{jl}\right)^2}{-2v_{\mathrm{j}}^2}\right] \\ &= \sum_{j=1}^{M} w_{kj}\exp\left(\frac{s_j}{-2v_{\mathrm{j}}^2}\right) \end{aligned}$$

The k-th output of RBFNN with perturbed input may be expressed as follow:

$$\begin{aligned} f_k(\mathrm{x}+\Delta\mathrm{x}) &= \sum_{j=1}^{M} w_{kj}\exp\left[\frac{\sum_{l=1}^{n}\left(x_l + \Delta x_l - u_{jl}\right)^2}{-2v_{\mathrm{j}}^2}\right] \\ &= \sum_{j=1}^{M} w_{kj}\exp\left(\frac{s_j^*}{-2v_{\mathrm{j}}^2}\right) \end{aligned}$$

Without any prior knowledge, unseen samples in S_Q are regard as having the same chance to appear. So, $\Delta\mathrm{x}$, whose elements is in $[-Q, Q]$,would be considered as input perturbations which are random variables having zero mean uniform distributions. Moreover, RBFNN stochastic model assumes that the inputs are independent and not identically distributed and weight perturbations are not considered. Here, we adopt the stochastic sensitivity measure in [1]. Considering the k-th output, the derivation is as follows and the detailed description can be found in [1].

$$\begin{aligned} & E_{S_Q}\left((\Delta y_k)^2\right) \\ &= E_{S_Q}\left[\left(\sum_{j=1}^{M} w_{kj}\exp\left(\frac{s_j^*}{-2v_{\mathrm{j}}^2}\right) - \sum_{j=1}^{M} w_{kj}\exp\left(\frac{s_j}{-2v_{\mathrm{j}}^2}\right)\right)^2\right] \\ &= \sum_{j=1}^{M}\left[\varphi_{kj}\left(\begin{aligned} &\exp\left(\frac{4\sum_{l=1}^{n}\sigma_{\Delta x_l}^2\left(\sigma_{x_l}^2 + (u_{x_l} - u_{jl})^2 + 0.2\sigma_{\Delta x_l}^2\right)}{2v_j^4} - \frac{2\sum_{l=1}^{n}\sigma_{\Delta x_l}^2}{2v_j^2}\right) \\ &-2\exp\left(\frac{\sum_{l=1}^{n}\sigma_{\Delta x_l}^2\left(\sigma_{x_l}^2 + (u_{x_l} - u_{jl})^2 + 0.2\sigma_{\Delta x_l}^2\right)}{2v_j^4} - \frac{\sum_{l=1}^{n}\sigma_{\Delta x_l}^2}{2v_j^2}\right) + 1 \end{aligned}\right)\right] \\ &\approx \sum_{j=1}^{M}\varphi_{kj}\left(\frac{\sum_{l=1}^{n}\sigma_{\Delta x_l}^2\left(\sigma_{x_l}^2 + (u_{x_l} - u_{jl})^2 + 0.2\sigma_{\Delta x_l}^2\right)}{v_j^4}\right) \\ &= \frac{1}{3}Q^2\sum_{j=1}^{M}\varphi_{kj}\left(E\left(S_j\right)/v_j^4\right) + \frac{0.2n}{9}Q^4\sum_{j=1}^{M}\varphi_{kj}/v_j^4 \end{aligned}$$

where $\varphi_{kj} = \left(w_{kj}\right)^2\exp\left(\frac{\mathrm{var}(s_j)}{2v_j{}^4} - \frac{E(s_j)}{v_j{}^2}\right)$

$$E(s_j) = \sum_{l=1}^{n}(\sigma_{x_l}{}^2 + (u_{x_l} - u_{jl})^2)$$

$$\mathrm{var}(s_j) = \sum_{l=1}^{n}\left(\begin{aligned} &E\left[\left(x_l - u_{x_l}\right)^4\right] - \sigma_{x_l}{}^4 + \\ &4E\left[\left(x_l - u_{x_l}\right)^3\right]\left(u_{x_l} - u_{jl}\right) + 4\sigma_{x_l}{}^2\left(u_{x_l} - u_{jl}\right)^2 \end{aligned}\right)$$

4.3. Minimizing TS

For a given Q, we compare the two trained classifier, f_1 and f_2. If f_1 has lower TS, it will have a better generalization capability. This is because the two classifiers use the same unseen samples to compute TS and f_1 can recognize them more effectively.

In this paper, we use the criterion function to construct RBFNN. We choose the RBFNN architecture which has minimum value of TS.

The new method of constructing RBFNN can be described as follows:

First, preprocess all samples. Normalize the input vector to [0,1] and output to a vector only including 0 and 1.

Let M be the number of hidden neuron centers.

Step 1: Set M to be the number of classes; Step 2: Use $\mathrm{k-means}$ clustering to find M hidden neuron centers; Step 3: Compute the width of hidden neuron centers and the connection weights; Step 4: For a given Q, compute the TS for the current RBFNN using (4); Step 5: $M = M + 1$. If the stopping criterion is not satisfied, go to Step 2; Step 6: Choose the RBFNN architecture which corresponds to the minimum TS.

Remark 1: In formula (4), we can enhance the effect of sensitivity by choosing Q with bigger value. Moreover, the samples in a neighborhood, which is decided by Q, should belong to the same class.

Remark 2: In Step 3, we set the widths of centers to be the multiple of the minimum distance between neighboring cluster centers [7] and get the connection weights using a pseudo-inverse method.

Remark 3: If M is too large, the trained RBFNN will not only be very complicated, but also over fit the training samples. Then its generalization capability is quite poor. To different dataset, We stop the search when M exceeds a value according to experience.

Remark 4: Furthermore, the criterion can be defined as

$$\begin{aligned} TS &= R_{emp} + E_{S_Q}\left((\Delta y)^2\right) \\ &= \lambda R_{emp} + (1-\lambda) E_{S_Q}\left((\Delta y)^2\right) \end{aligned}$$

where λ is a trade-off parameter to balance the effect of training error and stochastic sensitivity. This will be our future work.

5. Experimental results

The UCI Iris, Wine and Pima datasets are used to demonstrate the new method MTAS. Every dataset is divided into training set and testing set according to 7:3. This is repeated 10 times for generating 10 independent runs for each dataset.

The most critical issue in the construction of RBFNNs is to determine the number and position of hidden neuron centers. Our method add center on by one, and then choose the architecture with minimum TS. So, in this section, we compare the MTAS method with two other methods, whose number of centers is got by two ad-hoc estimation. The two methods get the hidden neuron centers using k-means clustering [8].
Method A: Use the square-root of the total number of samples to estimate the number of centers, and then search hidden neuron centers using k-means.
Method B: Consider the number of hidden neuron centers as the number of different output in the dataset, and then search hidden centers using k-means.

Table 1 shows some basic information of each dataset. There are maximum and minimum distances of normalized samples, the value of Q, and M bound where the search will stop, and so on.

Experimental results in different datasets are showed in Table 2, Table 3 and Table 4. The result analysis will be described later.

Datasets	Number of samples	Number of features	Number of classes	Maximum distance	Minimum distance	Q	M-bound
Iris	150	4	3	6.5105	0.1206	0.05	30
Wine	178	13	3	11.1800	1.1608	1	32
Pima	768	8	2	12.2052	0.3453	0.2	35

Table 1: Basic information of each dataset.

Methods	Average number of hidden neuron centers	Train accuracy mean (%)	Train accuracy stddv (%)	Test accuracy mean (%)	Test accuracy stddv (%)	TS
A	12	97.648	1.3229	96.447	1.8739	0.09132
B	3	84.408	2.0891	83.11	3.9468	0.19029
MTAS	11.1	97.357	1.0382	97.336	1.7512	0.07636

Table 2: Classification performance comparisons of different methods for UCI Iris dataset over 10 independent runs.

Methods	Average number of hidden neuron centers	Train accuracy mean (%)	Train accuracy stddv (%)	Test accuracy mean (%)	Test accuracy stddv (%)	TS
A	13	98.952	0.8534	98.335	1.6199	0.11304
B	3	96.853	0.5944	97.593	2.6270	0.15115

MTAS	11.8	98.577	0.8629	99.078	1.3081	0.10334

Table 3: Classification performance comparisons of different methods for UCI Wine dataset over 10 independent runs.

Methods	Average number of hidden neuron centers	Train accuracy mean (%)	Train accuracy stddv (%)	Test accuracy mean (%)	Test accuracy stddv (%)	TS
A	28	80.149	1.3822	75.281	2.8970	0.79395
B	2	66.871	2.4267	66.146	2.8622	11.63587
MTAS	20.9	78.398	1.4300	77.272	2.5028	0.30641

Table 4: Classification performance comparisons of different methods for UCI Pima dataset over 10 independent runs.

Analyzing the results, we find Method A is poor in training and testing accuracy. Because Method A uses less hidden centers and it could not describe the input vector space. Method B has better performance, but it has the tendency of over-fitting and more hidden neuron centers will lead to complicated RBFNN architecture. Moreover, the above two methods only give estimation on the number of hidden neuron centers. The trained RBFNNs using them are not stable and have higher values of TS. The proposed new method MTAS, which selects the RBFNN architecture by minimizing the sum of training error and sensitivity, gets a simple RBFNN. It gives the better training classification accuracy and the best classification accuracy in all the three datasets. Compared with the other methods, MTAS has smaller value of TS. The experimental results also show that the number of hidden neuron centers seriously influences the network's performance as in [9] and the appropriate hidden centers can lead to a better RBFNN architecture. The new method MTAS takes into consideration the information of training samples and unseen samples synchronously and it yields better RBFNN.

6. Conclusions and future work

In this paper, a new method MTAS for constructing RBFNN, based on training error and sensitivity, is proposed. The new method chooses the RBFNN which corresponds to the minimum TS. Here, TS is computed within a neighborhood of training samples. MTAS considers the generalization capability of RBFNN and it is unnecessary to ascertain the number of hidden centers in advance. Experimental results show that MTAS can select simple RBFNN architecture, which has better generalization performance. However, there are still some problems worthy of further study. We will further find a better condition to stop the search ahead. Moreover, we will consider the weighted sum of training error and stochastic sensitivity as the criterion function. It can balance the effect of the two parts more effectively. We will also use more datasets with different characteristics to demonstrate our new method.

Acknowledgement

This work is financially supported by National Science Fund of China (60473045) and Hebei Province High Technology project (04213533).

References

[1] W. Wing, D. S.YEUNG, D. F. Wang, E. Tsang and X. Z. Wang, Localized generalization error and its application to RBFNN training. *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, Guangzhou, 18-21 August 2005.

[2] G. B. Huang, P. Saratchandran and N. Sundararajan, A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. *IEEE Transactions On Neural Networks*, 16(1), January 2005.

[3] G. B. Huang, P. Saratchandran and N. Sundararajan, An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks. *IEEE Transactions On System, man, and Cybernetics*, 34(6), December 2004.

[4] H. Sarimveis, Alex Alexandridis and George Bafas, A fast training algorithm for RBF networks based on subtractive clustering. *Neurocomputing* 51:501-505, 2003.

[5] W. Wing, D. S.Yeung, X. Z. Wang and I. Cloete, A study of the difference between partial derivative and stochastic neural network sensitivity analysis for applications in supervised pattern classification problems. *Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai*, 26-29 August 2004.

[6] W. Wing, D. S. YEUNG and I. Cloete, Quantitative study on effect of center selection to RBFNN classification performance. *IEEE International Conference on Systems, Man and Cybernetics*, 2004.

[7] F. Schwenker, H. A. Kestler and G. Palm, Three learning phases for radial-basis-function networks. *Neural Networks* 14:439-458, 2001.

[8] M.M. Brizzotti et al, The influence of clustering techniques in the RBF networks generalization.

IEEProc. Of Conf. Of Image Processing and Its Applications, pp. 87-92, 1999.

[9] V. David and A. Sanchez, On the number and the distribution of RBF centers. *Neurocomputing* 7:197-202, 1995.