An adaptive RBF hidden layer generation algorithm

Haibo Chen¹, Chenyu Zhang²*, Gao Hu¹

¹School of Science, Zhejiang Sci Tech University, Hangzhou, China
²Essence Information Technology Co.Ltd, Building 4, No276, Huanghai Road, Tianjin, China

*Corresponding author e-mail: itouchzcy@163.com

Abstract. We present an adaptive learning algorithm of RBF neural network. This algorithm uses an adaptive splitting operation based on network sensitivity and sample density to dynamically change the number of nodes in the hidden layer of RBF network. At the same time, a refactoring operation based on energy consumption is proposed, and the connection weights of the hidden layer and the output layer are obtained by using the least square method. In the experiment of iris sample classification, the recognition rate of the model is 95%.

1. Introduction

[1] proposed a single hidden layer neural network, which uses radial basis function as the activation function of the central node of the network. The network is fast in convergence, can avoid local minimization [2], and was used widely in time series prediction, nonlinear classification and other problem [3][4].

There are many shortcomings in the practical application of RBF neural network. On the one hand, the accuracy of RBF neural network depends on the initial parameters. On the other hand, using k-means clustering to determine the distribution of hidden nodes often depends on expert experience. [5] used PSO to automatically adjust the center, variance and connection weight of each radial basis function. The automatically generated radial basis neural network (RBFN) of the algorithm can deal with the stabilization of inverted pendulum, the approximation of nonlinear function and the approximation of discrete dynamic system. [6] proposed a RBF neural network optimization algorithm, which chose a new method of mixed coding and synchronous optimization.

Based on bionics, we propose an adaptive method for constructing hidden layer nodes in RBF networks. This method is based on the principle of biological activity and energy consumption balance. When the energy is sufficient, the nodes continue to split to obtain better generalization. When the energy consumption is too large, the nodes begin to refactor to obtain a reasonable node distribution. The classification test shows that the method can achieve better classification accuracy.

This paper is organized as follows: the section 2 introduces the process of splitting and reconstructing the adaptive algorithm; the classification experiment is carried in section 4, and section 4 is the summary of the paper.
2. Self-adaptive learning algorithm

2.1. Definition

Definition 1: The fire rate of the sample vector $X_i$ at hidden node $j$.

$$V_{ij} = \exp\left(-\frac{\|X_i - C_j\|^2}{2\sigma_j^2}\right)$$ (1)

Definition 2: The total fire rate is sum of fire rates of all samples at all nodes

$$Ac = \sum_{i=1}^{p} \sum_{j=1}^{n} V_{ij}$$ (2)

Definition 3: Network energy consumption index

$$\text{Cec} = c\_k1 \times p + c\_k2 \times Ac / p$$ (3)

Where $c\_k1$ is the proportion parameter for calculating the number of energy consumption nodes. $c\_k2$ is the proportion parameter of energy distribution in the calculation of energy consumption.

Definition 4: Count of samples that can be differentiated (Dc).

When formula 4 is satisfied, it means that the new sample is distinguishable from the existing sample.

$$\|V_k - V_i\| > epl \times \max(1, \sqrt{n}) \quad i = 1, 2, \ldots, n$$ (4)

Where $V_k$ is the mapping vector of the new sample in the hidden space, $V_i$ is the mapping vector of the existing sample in the hidden space, and $epl$ is the sample coefficient.

Definition 5: Network sensitivity

$$Cs = \frac{Dc}{(p \times (p - 1)) / 2}$$ (5)

Definition 6: Central node variance

$$\sigma^2_i = \frac{W_{di}^2 \sqrt{m}}{4 \times \log(Sr)}$$ (6)

In which $W_{di}$ is the value range where the node release rate is greater than the width boundary.

Based on the above definition, we summarize the algorithm execution process into four steps:

The first step is to perform the splitting operation. According to the splitting operation, the central node of the network is expanded to form a network with approximation effect. The splitting operation considers network sensitivity and node sample density. Too large node sample density will make the node account for too large proportion in the final output, resulting in the error of this node has too great impact on the final result. However, the low sensitivity will reduce the number of distinguishable diversity, and can not achieve a good approximation effect.

The second step is the reconfiguration operation, which is responsible for merging too many network nodes. Too many nodes will seriously reduce the generalization effect of the network and improve the energy consumption. Therefore, when the energy consumption index of the whole network is higher than the threshold, the nodes will be merged to reduce the energy consumption.

The third step is to calculate the connection weight of the hidden layer and the output layer. The least square method is used to get the connection weight to minimize the sum of squares of errors.
2.2. Split strategy

The splitting strategy considers the sensitivity and sample density of the network. The sensitivity of the network is calculated first. When the sensitivity of the network is lower than the lower limit of the sensitivity or the sensitivity decreases continuously, the node splitting operation is performed. Secondly, the sample density of each central node is calculated. When the sample density is too large, the node splitting operation is also carried out. The split algorithm flow is as follows:

1) Calculate the network characteristic parameters: network sensitivity and node sample density.
2) For each sample input, calculate the divisible diversity between samples (the more divisible diversity between samples, the greater the sensitivity of the network; the less divisible diversity between samples, the lower the sensitivity of the network).
3) In order to ensure the linear separability of samples, we adopt a greedy idea and set a lower sensitivity limit. When the sensitivity of the network is lower than the lower sensitivity limit or the sensitivity decreases continuously, the node splitting operation is performed.
4) In order to make different features of input information reflected by different hidden layer neurons, we attribute the input samples to the nodes with the largest divergence rate. At the same time, in order to make the weak input information produce strong response near the smaller center, we calculate the sample density of each center node, and also split the nodes when the sample density is too large.
5) If the network node splitting operation is performed, the node to be split is found first, that is, the node with the highest total release rate of all nodes. The node to be split is divided into three sub nodes at most, and the calculation formula of the newly generated node center vector is as follows:

$$c_{newk} = c_{jk} - W_{dj}/2 + P_{o_{jk}} * W_{dj}/10 + W_{dj}/20$$

(7)

2.3. Refactoring strategy

Too many central nodes will affect the generalization ability of the network. In order to prevent the number of RBF nodes from exploding, we have formulated the corresponding reconstruction strategy to merge the central nodes to keep the network structure simple.

The process of refactoring strategy is as follows:

1) Calculate the average energy consumption and energy consumption index of the network.
2) Perform reconfiguration when the average energy consumption of the network is higher than the energy consumption index. The energy consumption index takes the number of nodes as an independent variable, and gets the maximum value when the number of nodes is equal to 5, that is, we expect the optimal number of network nodes to be 5. When the number of nodes is greater than 5, the more the number of nodes, the smaller the energy consumption index, and the easier the reconstruction operation is.
3) In the process of refactoring, the nodes with the smallest width should be found first. Then find the node closest to the node with the smallest width, and merge the node with the smallest width into the nearest node. The center vector and action width of the newly generated node are shown in formula 8 and formula 9.

$$C_{new} = (C_{min} + C_{nearest})/2$$

(8)

$$W_{new} = \max(W_{dmin}, W_{dnearest})/W_{p}$$

(9)

In which \(W_{p}\) is width scale factor.

3. Experiment

In this experiment, we selected iris as the sample of classification. The dataset contains three kinds of iris (iris setosa, iris virginica and iris versicolor), and each sample has four dimension attribute values.

The lower limit of sensitivity is 0.85, the proportion parameter of the number of energy consumption nodes is 0.5, the proportion parameter of the issued energy consumption is 0.5, and the lower limit of node sample density is 0.08.
At the end of all sample input, the network learning process ends. In the process of learning, the center node is divided 49 times and reconstructed 26 times, and 55 nodes are generated, including 30 leaf nodes. The leaf node is the central node of the final network.

In Figure 1, we can intuitively observe the evolution process of neural center nodes. When a node splits to generate a new node, the newly generated node is connected with it as a leaf node, and finally a node tree is formed. And each leaf node of the node tree is the network center node we need. The non leaf node in the graph is the process node generated in the process of network generation.

![Figure 1. Evolution process of network center node.](image)

After the distribution of hidden nodes is obtained, the connection weight of hidden layer and output layer is obtained by least square method. In order to test the performance of the generated neural network, the test set samples are input to the network for classification.

In the test set, we misjudged No84 samples from class II to class III, No134 and No135 samples from class III to class II, and all the class I samples were correctly classified. Thus, the recognition rate of network samples after learning is 95%.

4. Conclusion

The adaptive learning algorithm of RBF neural network proposed in this paper can quickly generate neural network hidden layer, which is different from the number of hidden nodes selected manually. A splitting strategy and a refactor strategy are adopted to generate center nodes.

Through the analysis of the experimental results of iris classification, The algorithm avoids the defect of setting the number of center vectors in advance, and has good applicability. The next research direction should consider using online method to dynamically modify the number of nodes, so as to obtain better learning effect.
References

[1] Park, J, Sandberg, I. W. Universal Approximation Using Radial-Basis-Function Networks. Neural Computation, 3(2):246-257.

[2] M. Safish Mary, V. Joseph Raj. Radial Basis Function Neural Classifier using a Novel Kernel Density Algorithm for Automobile Sales Data Classification. International Journal of Computer Applications, 2011, 26(6).

[3] Samuel Omar Tovias-Alanis, Wilfrido Gomez-Flores. Automatic construction of the complete architecture of a radial basis function network using differential evolution. 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE). IEEE, 2017.

[4] Gao M, Hong X, Chen S, et al. A combined SMOTE and PSO based RBF classifier for two-class imbalanced problems. Neurocomputing, 2011, 74(17):3456-3466

[5] Jia W, Zhao D, Shen T, et al. A new optimized GA-RBF neural network algorithm.[J]. Computational Intelligence & Neuroscience, 2014, 2014(3):44

[6] Chi F. Fung, Steve A. Billings, Wan Luo. On-line Supervised Adaptive Training Using Radial Basis Function Networks[J]. Neural Networks, 9(9):1597-1617.