Proceedings of the 2000 IEEE International Workshop
on Robot and Human Interactive Communication
Osaka, Japan - September 27-29 2000

Growing RBF Structures Using Self-organizing Maps

Qingyu XIONG, Kotaro HIRASAWA, Jinglu HU and Junichi MURATA
Dept. of Electrical and Electronic Systems Engineering, Kyushu Univ.
6-10-1 Hakozaki, Higashi-Ward, Fukuoka-City, Japan
(E-mail: xiong@terra.ees.kyushu-u.ac.jp)

Abstract

We present a novel growing RBF network structure using SOM in this paper. It consists of SOM and RBF networks respectively. The SOM performs unsupervised learning and also the weight vectors belonging to its output nodes are transmitted to the hidden nodes in the RBF networks as the centers of RBF activation functions, as a result one to one correspondence relationship is realized between the output nodes in SOM and the hidden nodes in RBF networks. The RBF networks perform supervised training using delta rule. Therefore, the current output errors in the RBF networks can be used to determine where to insert a new SOM unit according to the rule. This also makes it possible to make the RBF networks grow until a performance criterion is fulfilled or until a desired network size is obtained. The simulations on the two-spirals benchmark are shown to prove the proposed networks have good performance.

1 Introduction

The main difficulty faced in the field of feedforward neural networks (FNN's) is model selection. Model selection involves matching the complexity of the function to be approximated with the complexity of the model. FNN model complexity is determined by the factors such as the number of weights, the values of weights, and weight connection topology. If the model does not have the enough complexity to approximate the desired function, underfitting and poor generalization occur. If the model is too complex, then it may overfit the data and also give poor generalization.

In one group of the model selection techniques, training begins with an oversize network which is then simplified. Pruning is one of such techniques[1]. The nonconvergent technique of early stopping[2] also uses an oversize network, and works by stopping the training at a point where the performance for a validation set begins to worsen. Since the validation set is a representative of the underlying function, the performance for this data set will worsen when a model is formed, which is more complex than the underlying function.

But the main disadvantage is the difficulty of specifying what size of the network should actually be considered as an oversize network a priori. If the initial network selected is too small, it will be unable to have a good solution and hence underfit the data. On the other hand, selecting an initial network that is much larger than required makes training computationally expensive.

Another group of the model selection techniques uses constructive methods[3]. These methods start with a minimal size network, and sequentially add units according to some criterion until an appropriate level is reached. Constructive algorithms will spend the majority of their time on training networks smaller than the final network, as compared to the algorithms that start training with the oversize networks. There are a number of inherent advantages in constructive algorithms over the algorithms using the oversize networks. But one must define the type of training algorithms and how to connect the new hidden nodes to the current networks.

Generally speaking, Self-organizing neural network models generate mapping from high-dimensional signal spaces to lower-dimensional topological structures[4]-[6]. These mapping are able to preserve the neighborhood relations in the input data. Most of the works carried out so far on SOM's have concentrated on systems with a single self-organizing layer. This self-organizing layer has generally a fixed number of neurons. However in recent years, some adaptive/multilayer SOM models have also been proposed. Lee *et al.*[7]proposed a self-development neural network for adaptive vector quantization. The network has one self-organizing layer and two levels of adaptations–namely structure and parameter (synaptic vector) levels. Kohonen's topology preserving mapping algorithm was used for parameter adaptation. Wu *et al.*[8] investigated a supervised two self-organizing layer SOM. However their method did not employ any structure with adaptation scheme.

Cho [9] proposed a structure-adaptive SOM with a single self-organizing layer. Bauer *et al.*[10]presented a growing self-organizing map (GSOM) algorithm. The GSOM has generated a hypercubical shape which is adapted during the learning. Fritzke [11]proposed a structure adaptation algorithm for SOM which has one self-organizing layer with sophisticated multidimensional lattice topologies. The algorithm includes cell insertion and removal based on the local counter variable.

0-7803-6273-X/00/$10.00©2000 IEEE

According to these, we propose a network named growing RBF using SOM in this paper, which is composed of two kinds of networks such as a basic network and a cluster network. In such network's structure, a RBF network and a Kohonen SOM network are adopted as the basic network and cluster network, respectively (see Fig.1). The relationship between the output nodes in SOM and the hidden nodes in the RBF network is one to one correspondence.

2 Structure of Growing RBF Network Using SOM

2.1 Basic Structure

Let $R = (r_1, r_2, \cdots, r_N)$ denote the input vector, and Y is the output of the basic network. Each node in the output layer of SOM has a weight vector, say W_l, attached to it, where $W_l = (w_{l1}, w_{l2}, \cdots, w_{lN})$ $_{l \in L}$. Also W_l is transmitted to the hidden nodes in the RBF network as its center of RBF activation function. The usual procedure for training such a network consists of two consecutive phases, an unsupervised and a supervised one.

This network starts with a minimal size network, and then sequentially adds neurons according to the criterion. SOM network performs unsupervised learning. It generates ordered mappings of the input data onto some low-dimensional topological structure, which means that the SOM defines a topology mapping from the input data space onto the output nodes. Then the weight vector W_l belonging to the output in SOM are transmitted to the hidden node in the RBF network as its center of RBF activation function.

The RBF network performs supervised training using delta rule. The current output errors in the RBF network are used as the values of the current best-matching unit in SOM to determine where to insert a new SOM unit according to a rule. This also makes it possible to make the RBF network grow until a performance criterion is fulfilled or until a desired network size is reached, because of their connecting relationship between the output nodes in SOM and the hidden nodes in the RBF network.

2.2 Center adaptation

When an input vector R is given to this network, at first in SOM, the Euclidean distances between each W_l and R are computed. The output nodes of SOM compete each other, but the node c whose connected weight vector is closest to the current input vector (minimum distance) wins, say best-matching unit.

$$c = arg\ Min_{\ l \in L}\ \{\| R - W_l \|\} \tag{1}$$

After determining the best-matching unit for the current input pattern, this best-matching unit and its topological neighbors should be increased according to the principle proposed by Kohonen[4]-[6]. There are, however, two differences:

- The adaptation strength is constant over time. Specifically, we use constant adaptation parameters α_c and α_n for the best-matching unit and its neighboring units, respectively.
- Only the best-matching unit and its first topological neighbors are adapted.

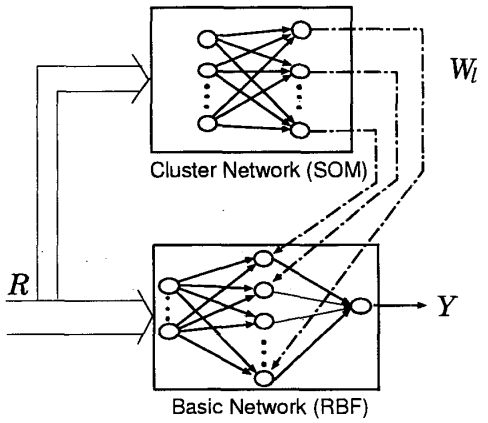

Figure 1: Structure of Growing RBF Network using SOM

Here, let N_c denotes the set of the first topological neighbors of the winner c.

An update rule in SOM can be formulated as follows:

$$\begin{cases} W_c \leftarrow W_c + \alpha_c * (R - W_c) & \\ W_l \leftarrow W_l + \alpha_n * (R - W_l) & (\text{for } l \in N_c) \\ W_l \leftarrow W_l & (\text{otherwise}) \end{cases} \tag{2}$$

The inputs of the basic network is the same as the ones in SOM. Each Gaussian unit l has an associated vector W_l indicating the center in the input vector space and the standard deviation σ_l defined by using the mean distance of its first neighboring Gaussian units.

For a given input vector R, the activation function of unit l in RBF is described by:

$$f_l(R) = exp(-\frac{\|R - W_l\|^2}{\sigma_l^2}) \tag{3}$$

The Gaussian units are completely connected to the linear output node by weighted connection w_l. Therefore the value of the output node in the RBF network is computed by:

$$Y = \sum_{l \in L} w_l f_l \tag{4}$$

2.3 Local error measure

The weight w_l of the output layer is trained to produce the desired value T at the output node. Common delta rule is used.

$$w_l \leftarrow w_l + \beta(T - Y)f_l \quad (\text{for all } l \in L) \tag{5}$$

where β is the learning rate.

The error caused by the training data is used to determine where to insert a new unit in the input space. Here for each unit l in SOM, we define a local counter variable V_l basically containing the errors of the node where the best-matching for the current input is obtained.

$$V_l \leftarrow \begin{cases} V_l + |T - Y|^2 & \text{for } l = c \\ V_l & \text{otherwise} \end{cases} \tag{6}$$

2.4 Insertion of new cell

After a fixed number of adaptation steps M in SOM and RBF network, we determine the unit s with the largest values of V_l:

$$s = arg\ Max_{\ l \in L}\ \{V_l\} \tag{7}$$

Then, we look for a unit with the largest error counter value among the first neighbors N_s. This is a unit f satisfying:

$$f = arg\ Max_{\ l \in N_s}\ \{V_l\} \tag{8}$$

And in those common neighbors $N_{s \bigcap f}$ of s and f, we determine the cell u with the largest distance from cell s:

$$u = arg\ Max_{\ l \in N_{s \bigcap f}}\ \{||W_l - W_s||\} \tag{9}$$

After that, we now can insert a new unit I between s and u. The position of a new unit I in the input space is initialized as:

$$W_I = (W_s + W_u)/2 \tag{10}$$

And to keep topological relationship in SOM, we should set s, f and those common neighbors $N_{s \bigcap f}$ as I's first neighbors. Then, the original neighboring relationship between s and f is relevantly cancelled.

Doing so, we may say that the local optimum regarding SOM structure and basic network's output error is realized. This is due to the fact that new units are only inserted in those regions of the input space where misclassifications still occur and the errors are locally maximum.

2.5 Determining of initial weight about the new cell

Whenever a new unit I is inserted, a weight w_I connecting to the output node in RBF network should be determined. Here, we make initial w_I equal to 0 simply not to make influence to the output value of RBF network after inserting.

2.6 Stopping criterion and the whole learning process

To prevent the network growing indefinitely, a stopping criterion has to be defined. In the simplest case one can specify a maximum number of allowed units and halt the process when this number is achieved or surpassed.

The whole process of the learning of the proposed network is arranged as follows:

- Initialize the SOM's structure.
- Create a weighted connection from each RBF cell to the output unit in the RBF network.
- Associate every cell in the RBF network with a Gaussian function.
- while (an appropriate criterion is not reached)
  - repeat λ times
    - Choose I/O-pair from training data;
    - Determine best-matching unit c (see eq.1);
    - Increase matching for c and its first neighbors in SOM (see eq.2);
    - Compute activation f_l (see eq.3);
    - Compute the output Y (see eq.4);
    - Perform one step delta-rule learning for the weights w_l (see eq.5);
    - Add local counter variable V_l (see eq.6);
  - Determine cell s with maximum local counter value (see eq.7);
  - Determine cell f with maximum local counter value among sth cell's first neighbors N_s (see eq.8);
  - Determine cell u with the largest distance between s and the common neighbors $N_{s \cap f}$ (see eq.9);
  - Insert a new cell I between s and u (see eq.10);
  - Give the new cell I an initial weight connecting to the output in the RBF network;
- Stop if pre-specified conditions are met.