Sequential Bayesian Learning for Modular Neural Networks

Pan Wang[1], Zhun Fan[2], Youfeng Li[1], and Shan Feng[3]

[1] Wuhan University of Technology, Wuhan 430070, China
{Wang,Li,jfpwang}@tom.com
[2] Technical University of Denmark
zf@mek.dtu.dk
[3] Huazhong University of Science and Technology, Wuhan 430074, China

Abstract. In this paper, we present a distributed computing method, namely Sequential Bayesian Learning for modular neural networks. The method is based on the idea of sequential Bayesian decision analysis to gradually improving the decision accuracy by collecting more information derived from a series of experiments and determine the combination weights of each sub-network. One of the advantages of this method is it emulates humans' problems processing mode effectively and makes uses of old information while new data information is acquired at each stage. The results of experiments on eight regression problems show that the method is superior to simple averaging on those hard-to-learn problems.

1 Introduction

Modular neural network is an effective kind of connectionism models that consists of a group of sub-networks combined to solve complex problems. It often produces superior results than single well-trained neural network does. It has been a hot topic in many areas such as pattern recognition and classification, image processing, system identification, language/speech processing, control, modeling, target detection/recognition, fault diagnosis, etc.

Here, we shall use the term modularity in the widest meaning, that is, modular neural network is a system composed of a group of neural networks, which are independent, inter-connected, co-operative in structure level or in function level. The basic unit in this system is a module. Therefore, in the literature the paradigms such as multiple neural networks, hybrid neural networks, distributed neural networks and committee machine could be unified under the aforementioned framework. In this meaning, we give corresponding architecture and description:

$$\text{MNN}=\langle X, \text{C}, \text{SN}, \text{IU}, Y\rangle \tag{1}$$

Where, $X\in D \subseteq R^n$, is the input vector; C represents a classifier whose function is to decompose input space or I/O space in the system; SN represents a set of subnets $\{Net_i\}_{i=1}^{K}$; IU represents the integrating unit which performs adaptive combination of modules; $Y \in E \subseteq R^m$, is the output vector. The corresponding network architecture is illustrated in Fig.1

In addition, if the classifier doesn't work (i.e. the input or I/O space is classified into one class), the corresponding modular neural network is named as "neural net-

J. Wang, X. Liao, and Z. Yi (Eds.): ISNN 2005, LNCS 3496, pp. 652–659, 2005.

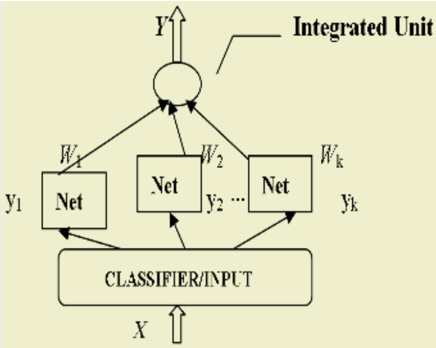

Fig. 1. Network Architecture.

work ensemble"; If any Net_i is composed of some sub-nets, the corresponding modular neural network should be named as "hierarchical modular neural networks". In this paper, we focus on the "neural network ensemble".

In the past few years, one of the research directions on modular neural network is how to combine the outputs of the component networks to form the output of the entire system that has the best performance. In the context of regression problems, diverse linear combination methods, such as simple averaging [1], MLS-OLCs [2], fuzzy integral [3], etc, are presented to integrate the component networks. Whatever any specific linear combination method is used, if K neural networks arc selected to form an entire system, the outputs of the component networks are then combined through weighted sum where a combination weight $w_i (i = 1,2,..., K)$ is assigned to the i-th component network. Consequently, the output vector $\overline{y}$ of the entire system is determined according to Eq. (1) where y_i is the output vector of the i-th component network.

$$\overline{y} = \sum_{i=1}^{K} w_i y_i \tag{2}$$

For simplicity, here we assume that each component network has only one output variable, i.e. the function to be approximated is $f : R^m \rightarrow R$. It can be easily generalized to the situations where each component network has more than one output variable.

Note that the combination weights are different from the connection weights belonging to a specific component network. The former are inter-network connection coefficients in the whole system while the latter are intra-network connection coefficients in the corresponding component network. Particularly, the sum of the combination weights are constrained to unity and each combination weight is set to be positive, that is,

$$\sum_{i=1}^{K} w_i = 1 \tag{3}$$

$$w_i \geq 0 \tag{4}$$