Formation and dynamics of modules in a dual-tasking multilayer feed-forward neural network

Chi-Hang Lam and F. G. Shin
Department of Applied Physics, Hong Kong Polytechnic University, Hung Hom, Hong Kong
(Received 17 April 1998)

We study a feed-forward neural network for two independent function approximation tasks. Upon training, two modules are automatically formed in the hidden layers, each handling one of the tasks predominantly. We demonstrate that the sizes of the modules can be dynamically driven by varying the complexities of the tasks. The network serves as a simple example of an artificial neural network with an adaptable modular structure. This study was motivated by related dynamical nature of modules in animal brains. [S1063-651X(98)14409-4]

PACS number(s): 87.10.+e, 02.50.−r, 05.20.−y

I. INTRODUCTION

Training of neural networks for complicated tasks is often difficult due to the large number of local minima in the error function landscapes [1,2]. There has been much effort in searching for efficient learning algorithms and better network architectures. An interesting idea put forward by Jacobs *et al.* is the use of modular neural networks [3]. In this approach, a complicated task is broken down into several simpler subtasks. The whole network consists of modules called expert networks each of which only learn to solve one of the subtasks. Their outputs are connected to the overall network output via a gate network. The duty of the gate network is to select which expert network is to be consulted for any given input pattern. The expert and the gate networks are trained simultaneously to achieve coherently the inter-related processes of task decomposition, assignment of subtasks to the modules, and the actual learning of the task. Improved learning algorithms for modular neural networks and examples of applications to control problems and speech recognition are discussed in Ref. [4]. Apart from artificial neural networks, animal brains also have high level modular structures controlling various body functions [5]. At a lower level, it has also been suggested recently that the human brain adopts a modular decomposition strategy to learn sets of visuomotor maps for relating visual inputs to motor outputs under various conditions [6].

To construct a modular artificial neural network, both the overall modular structure and the architecture of the individual components in general have to be carefully defined before training commences. Spontaneous formation of modules from a homogeneous network has also been investigated by Ishikawa [7]. The author adopted a weight decay training approach. A group of neurons is considered to have formed a module when most connections to other groups of neurons have decayed away. Once formed, the modular structure remains unchanged.

On the contrary, regions in an animal brain responsible for various body functions are found to have certain fluidity even for adults. In an experiment by Fox [8], regions in a monkey's brain responsible for various fingers were mapped. A middle finger was then amputated. After a few weeks, the regions corresponding to adjacent fingers surprisingly expanded into the region previously controlling the middle finger. This result indicates that fine details in the modular structure of an animal brain are not hard coded genetically and some rezoning may be allowed to adapt to the changing environment or conditions.

Ideally, a modular neural network with a dynamical architecture may offer enhanced performance. It may automatically decompose a complex task into smaller ones with the least design effort but the best adaptability in a continuously changing environment. To some extent, an animal brain may be an example. Brains are by far the most powerful neural networks and often inspire advances in their artificial counterparts. Motivated by the fluidity of their modular structures, we have in this work constructed and studied an artificial neural network that exhibits analogous adaptable modular structures. In our network, modules are formed upon training and their sizes change when the complexities of the associated tasks vary with time. In the investigations of both Jacobs *et al.* and Ishikawa, the modular structure of the neural networks remains unchanged once they are defined or generated. Our construction is to our knowledge the first example of a neural network with an adaptable modular architecture. However, at present, the dynamics only occur for some specific tasks, network architectures and training parameters. We also obtain no improvement in the efficiency of training. Therefore, our results may be of limited immediate practical interest.

In Sec. II, we specify the architecture of our neural network, the tasks to be learned, and the training method. Section III explains the formation of modules. Section IV describes their dynamical properties when the complexities of the tasks become time dependent. We conclude in Sec. V with some further discussion.

II. NETWORK ARCHITECTURE AND TRAINING

We focus on a multilayer feed-forward neural network for function approximations. The network architecture is shown in Fig. 1. The circles denote the neurons and the lines represent the synaptic connections. The network is composed of an input layer, three hidden layers, and an output layer. Note that the neurons in a hidden layer are only connected to neighboring ones in the next layer. This makes the locations of the neurons physically significant and is essential for the generation of any spatial modular pattern. The values received by the two input neurons are denoted by ξ_1 and ξ_2,

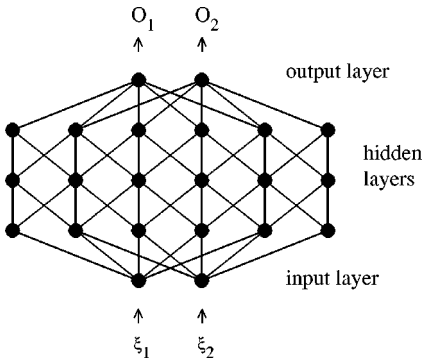

FIG. 1. Architecture of a feed-forward network for approximations of two independent functions.

respectively. The output $V_i^{(1)}$ of the ith neuron in the first hidden layer is given by

$$V_i^{(1)} = \tanh\left[\sum_{j=1}^{2} w_{ij}^{(0)} \xi_j + \theta_i^{(0)}\right], \tag{1}$$

where $w_{ij}^{(0)}$ and $\theta_i^{(0)}$ are weight and bias, respectively. The outputs of the neurons in the second and the third hidden layers corresponding to $m=2$ and 3, respectively, are given by

$$V_i^{(m)} = \tanh\left[\sum_{j=1}^{6} w_{ij}^{(m-1)} V_j^{(m-1)} + \theta_i^{(m-1)}\right]. \tag{2}$$

The weight $w_{ij}^{(m-1)}$ is zero if the corresponding synaptic connection does not exist. The overall network outputs O_1 and O_2 are computed from

$$O_i = \sum_{j=1}^{6} w_{ij}^{(3)} V_j^{(3)} + \theta_i^{(3)}. \tag{3}$$

This network is capable of approximating an $\mathbb{R}^2 \to \mathbb{R}^2$ function. However, we limit our consideration to the particular case of approximating two $\mathbb{R} \to \mathbb{R}$ functions. Specifically, O_1 and O_2 are supposed to approximate two target functions $T_1(\xi_1)$ and $T_2(\xi_2)$, respectively. Furthermore, we consider the case in which ξ_1 and ξ_2 are uncorrelated inputs. The target function $T_1(\xi_1)$ is thus completely independent of ξ_2 and similarly $T_2(\xi_2)$ is independent of ξ_1. Therefore, the problem has a natural decomposition into two uncorrelated subtasks of function approximations.

All target functions studied in this work belong to a family of sawtooth functions defined by

$$f_r(\xi) = h^4[\tfrac{1}{2}(\xi + r - 1)], \tag{4}$$

where h^4 denotes the fourfold composite $h(h(h(h(x))))$ of the function $h(x) = |rx - 1|$. The complexity of f_r depends strongly on the parameter r. Figure 2 shows some examples. At $r=1$, it is a simple straight line. The complexity increases monotonically with r until it becomes a complicated sawtooth function at $r=2$.

The weights and biases in the network are first initialized to random values. Training is conducted using a backpropagation algorithm in on-line mode with a momentum term of 0.9 [1]. In this approach, a set of input values ξ_1 and ξ_2 are chosen randomly and independently in the range $0 \leqslant \xi_i \leqslant 1$ at each time step. The input pattern is then presented to the network and the resulting output error guides a single step of adjustment on every weight $w_{ij}^{(m)}$ and bias $\theta_i^{(m)}$. A similar adjustment is made during every time step using a random input pattern. This approach is effectively a steepest descent method minimizing the output error

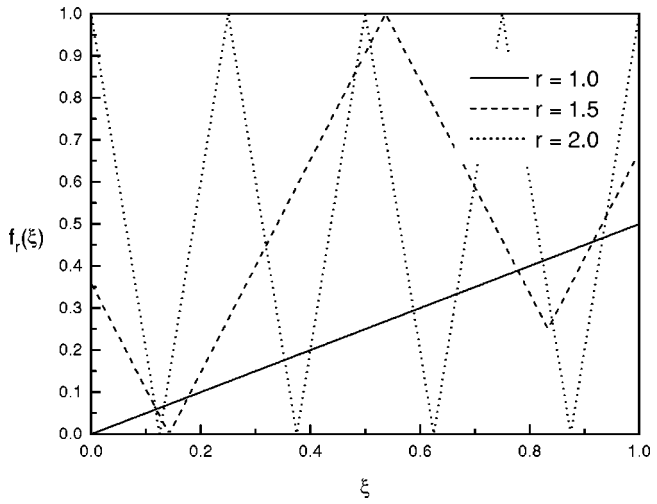

FIG. 2. Family of sawtooth function f_r for $r=1.0$, 1.5, and 2.0. The complexity of the function increases with r.

$$E = \langle (O_1 - T_1)^2 + (O_2 - T_2)^2 \rangle. \tag{5}$$

The brackets denote averaging over all input patterns. It is customary to introduce noise to the training process to avoid local minima [1]. In our case, random excitations are applied to inactive neurons to recover their activities. Specifically, we randomly inspect a random neuron in the hidden layers at each time step. The neuron is identified as inactive if either its recent outputs have a rms fluctuation smaller than 0.3 or the sum of the magnitudes of all its connections is smaller than 0.15. An inactive neuron is excited by updating all immediately associated weight $w_{ij}^{(m)}$ to $0.98 w_{ij}^{(m)} + \eta_{ij}^{(m)}$ where $\eta_{ij}^{(m)}$ is a uniform random variable in the range ± 0.2.

III. MODULES FORMATION

We now discuss the process of modularization of our neural network during training. Identical target functions $T_1 = T_2 = f_r$ with $r=1.5$ are considered. We have already explained in Sec. II that approximations of the functions are uncorrelated tasks since the inputs are independent. The sawtooth function f_r defined in Eq. (4) is moderately complicated for $r=1.5$. We perform backpropagation training until time $t = 3 \times 10^6$, well after the output errors have converged apart from some random fluctuations. Figure 3(a) shows the resulting network in a typical run. The intensity of a line representing a connection is proportional to the absolute value of the corresponding weight. In the figure, weights of magnitude less than 0.05 are invisible while those larger than 1 are completely darkened. Some allowed connections have practically vanished so that the network clearly decomposes into two modules. Neurons are represented by open or filled

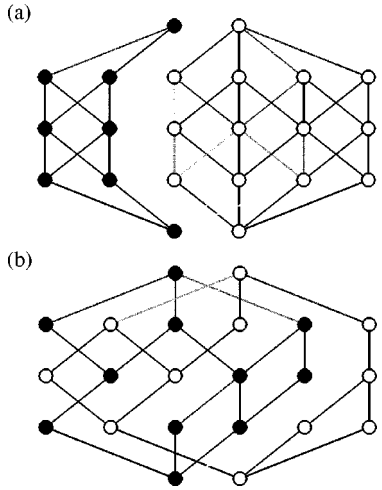

FIG. 3. Two realizations of a network trained to approximate two identical functions with independent inputs.

circles depending on which module they belong to. In this particular run, the modules contain 6 and 12 hidden neurons, respectively. The performance of a module can be evaluated from the error ϵ_1 or ϵ_2 given by

$$\epsilon_i = \langle |O_i - T_i| \rangle, \tag{6}$$

where the averaging is over all input patterns. We obtain $\epsilon_1 = 0.034$ and $\epsilon_2 = 0.008$ in this run. Since the two tasks are identical, a larger module usually gives a smaller error as is observed here.

Figure 3(b) shows another realization trained under the same conditions. In this case, the modules are noncompact in contrast to the compact ones obtained previously. Both modules have 9 hidden neurons and the output errors are $\epsilon_1 = 0.042$ and $\epsilon_2 = 0.010$, respectively. In fact, noncompact modules are not favored energetically since they usually give slightly larger error due to the fewer internal connections. They are formed occasionally because of entropic reasons.

IV. DYNAMICS OF MODULES

We now demonstrate that the modules in our neural network can expand or shrink if the associated tasks vary with time . The target functions are set to be $T_1 = f_{r_1}$ and $T_2 = f_{r_2}$ in the same family defined in Eq. (4). The parameters r_1 and r_2 are given by

$$\begin{aligned} r_1 &= 1.5 + R \sin\theta, \\ r_2 &= 1.5 - R \sin\theta, \end{aligned} \tag{7}$$

where $\theta = (2\pi t/T)$ modulo 2π is a time dependent phase angle and R and T are the amplitude and period of the variation, respectively. Training proceeds continuously while the tasks are slowly varying since one backpropagation step is executed at each time step. The periodic variation in the tasks hence directly implies periodic changes in the weights and biases of the network.

We first consider an amplitude $R = 0.4$ and a period $T = 5 \times 10^6$. This is a rather long period and the system is not far from the quasistatic limit. We discard any data for time

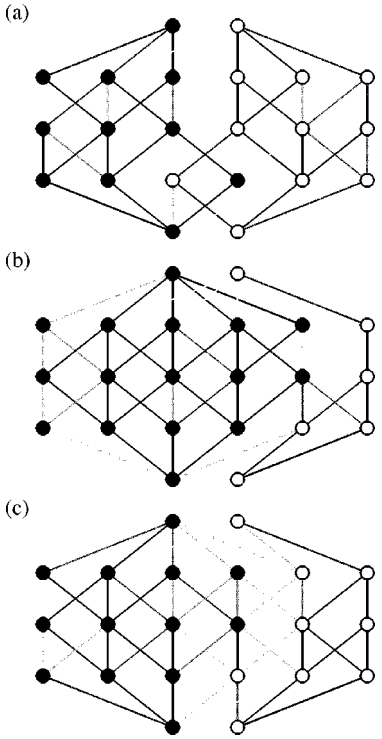

FIG. 4. Realizations of a network trained to approximate time-dependent functions at phase angles (a) $\theta = 0$, (b) $\theta = \pi/2$, and (c) $\theta = \pi$.

$t < 2T$ to avoid any initial transient. Figure 4 shows snapshots of the network at three different instants in the same period in a typical run. Two modules are formed, similar to the static case discussed in Sec. III, but they now expand and shrink continuously. In all three snapshots, there are neurons in the process of switching from one module to another and hence the modules are not completely decoupled. We therefore introduce a more objective criterion for associating the neurons with the modules, which is consistent with the previous classifications by simple inspection. We define that a neuron is in a module if its output has a stronger effect on the overall output of this module than on the other one. For example, the *i*th neuron in the *m*th layer belongs to module 1 if

$$\left\langle \frac{\partial O_1}{\partial V_i^{(m)}} \right\rangle > \left\langle \frac{\partial O_2}{\partial V_i^{(m)}} \right\rangle. \tag{8}$$

We observe that the modules are more compact in the dynamical case. This clearly results from the dynamics of gradual expansion and shrinkage of the modules.

The snapshot in Fig. 4(a) is at phase angle $\theta = 0$. From Eq. (7), the parameters are $r_1 = r_2 = 1.5$, implying tasks of identical complexity at that instant. Let n_1 and n_2 be the number of hidden neurons in the respective modules. We obtain $n_1 = n_2 = 9$ and individual output errors $\epsilon_1 = 0.040$ and $\epsilon_2 = 0.014$. The situation here is quite similar to the static case. Figure 4(b) shows the state of the network a quarter of a period later at $\theta = \pi/2$. Now, $r_1 = 1.9$ and $r_2 = 1.1$. Module 1 on the left is handling a much more complicated function approximation task and has already gained some neurons from module 2, which in contrast is assigned a much simpler

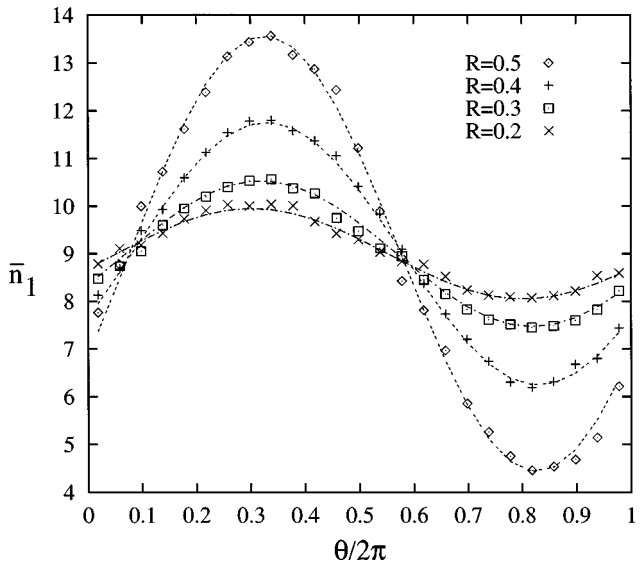

FIG. 5. Plot of the average size n_1 of module 1 against phase angle θ for various values of the amplitude R at period $T=5\times10^6$.

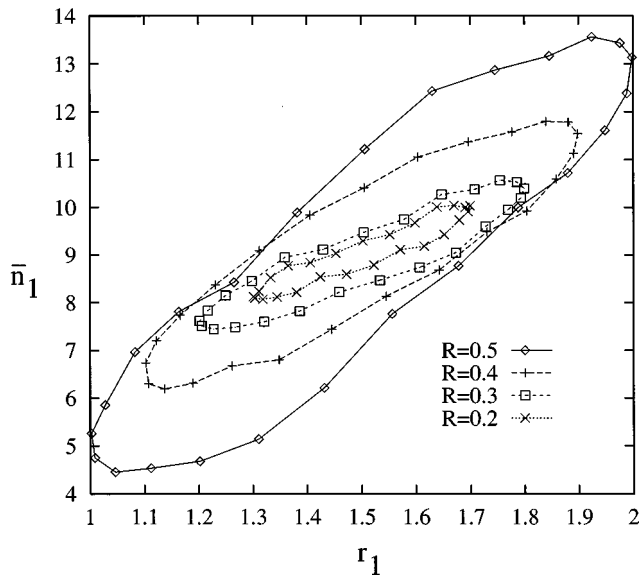

FIG. 6. Plot of the average size n_1 of module 1 against the target function parameter r_1 for various values of the amplitude R.

task. We get $n_1=14$, $n_2=4$, $\epsilon_1=0.075$, and $\epsilon_2=0.007$. Even though module 1 is larger, its corresponding error is significantly higher because of the more complicated target function to be approximated. The large difference between the errors is precisely the driving force for the redistribution of the neurons. At $\theta=\pi$, the tasks become identical again. The corresponding state shown in Fig. 4(c) indicates modules of sizes $n_1=11$ and $n_2=7$, respectively. The errors are $\epsilon_1=0.011$ and $\epsilon_2=0.064$. Module 1 has already returned most neurons to module 2 but is still retaining a few extra ones. This exemplifies a hysteresis effect when tuning the module sizes with the complexities of the tasks. Note that hysteresis also exists for other values of θ but is not apparent, for example, in Fig. 4(a) due to the presence of random fluctuations.

We now examine quantities averaged over time. Let $\overline{n_1}$ be the size of module 1 at any given phase angle θ averaged during the period $2T<t<100T$. The average size of module 2 is then $18-\overline{n_1}$. Figure 5 plots $\overline{n_1}$ against θ for various values of the amplitude R. The relations between $\overline{n_1}$ and θ fit quite well to sinusoidal functions in the form $9+A\sin(\theta-\phi)$, which are also plotted in Fig. 5. The values of the phase shift ϕ fall in a rather narrow range of 0.40 ± 0.07. This phase shift measures the lag of the variation in the module sizes behind that of the complexities of the tasks. We present the same data again in Fig. 6 in an $\overline{n_1}$ versus r_1 plot showing the hysteresis loops. For $R\lesssim0.1$, the variations in the tasks become so small that the modules become static. Other values of the period T are also investigated. Figure 7 shows a similar plot of $\overline{n_1}$ versus θ for $R=0.4$ and T varying from 5×10^5 to 2×10^7. At small T, the relations deviate significantly from sinusoidal.

V. DISCUSSION

To construct the above neural network exhibiting dynamical modules, we have been very careful in selecting the network architecture, the tasks, and the training method. Although some variations are allowed, the dynamics unfortunately is not robust but we have identified some essential characteristic features. In the network architecture we used, only neighboring neurons in the hidden layers are connected as has been explained in Sec II. Neurons on the surface of a module thus have fewer connections and are effectively weakly bonded. Therefore, an expanding module can easily capture neurons on the surface of the other module one by one. For the more widely used architecture with full connections between neurons in adjacent hidden layers, the notion of surface or bulk does not apply. It is much more difficult to set loose some individual neurons for reallocation without dissolving the whole module. The choice of the target functions is also important. They have to be sufficiently complicated so that it requires as many neurons as possible for the computations. However, it cannot be too complicated

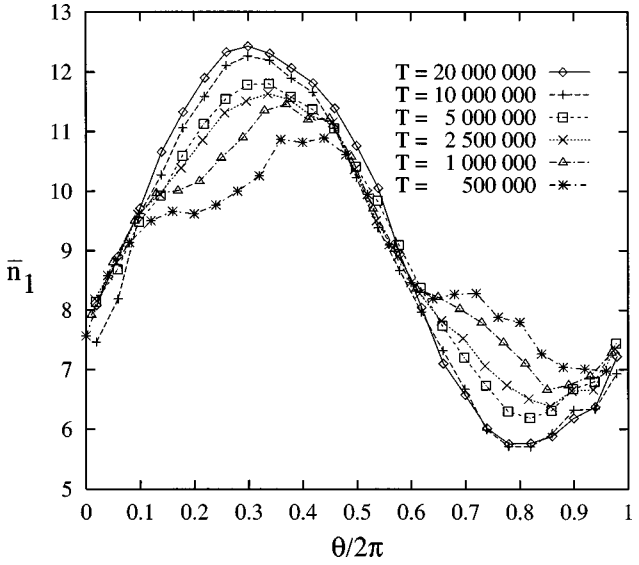

FIG. 7. Plot of the average size n_1 of module 1 against the phase angle θ for various values of the period T at amplitude $R=0.4$.

because the abundance of local minima would forbid efficient learning. We have found that the family of sawtooth functions defined in Eq. (4) suits this purpose nicely. We also tried sinusoidal target functions as examples but the resulting dynamics of the modules is less pronounced.

We have obtained sinusoidal relations between the module size $\overline{n}_1$ and the phase angle θ of the variation of the tasks as shown in Fig. 6. The simplicity of these relations is particularly interesting, although it does not hold for smaller periods or some other target functions that we checked. Using Eq. (7) and neglecting the hysteresis effect, the sinusoidal relations imply $\Delta n \propto \Delta r$ where $\Delta n = n_2 - n_1$ and $\Delta r = r_2 - r_1$. Because the complexities of the tasks we used increase monotonically with r, we may tentatively identify r_i with some complexity measure c_i for the respective modules. As a result, we can write $\Delta n = \kappa \Delta c$ where $\Delta c = c_2 - c_1$. The proportionality constant κ is related to the compressibility of the modules with respect to changes in the complexities of the tasks. The application of concepts in thermodynamics motivated by the above observations may be helpful for further investigations.

We have studied a network with separate input and output neurons for each task. It would be more interesting if the tasks could share the same input and output nodes but appropriate information could be channeled automatically to the correct module similar to the networks of Jacobs *et al.* [3] and Ishikawa [7]. However, we have not yet been able to construct such a system exhibiting both spontaneous modularization and module dynamics.

In conclusion, motivated by the fluidity of modules in the brain, we have proposed a novel artificial neural network with analogous adaptable modular structures. When training a network to perform two independent function approximation tasks, two corresponding modules are formed. Their sizes can vary in order to adapt to variations in the complexities of the tasks. Hysteresis in the dynamics is observed and compactness of the modules is enhanced by the process of expansion and shrinkage. We have also discussed features in our model that are essential for the dynamics.

ACKNOWLEDGMENT

This work was supported by RGC Grant No. 0354-046-A3-110.

[1] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation* (Addison-Wesley, New York, 1991).
[2] T. L. H. Watkin and A. Rau, Rev. Mod. Phys. **65**, 499 (1993).
[3] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, Neural Comput. **3**, 79 (1991).
[4] M. I. Jordan and R. A. Jacobs, in the *Handbook of Brain Theory and Neural Networks*, edited by M. Arbib (Cambridge University Press, Cambridge, 1995).
[5] P. Peretto, *An Introduction to the Modeling of Neural Networks* (Cambridge University Press, Cambridge, 1992).
[6] Z. Ghahramani and D. M. Wolpert, Nature (London) **386**, 392 (1997).
[7] M. Ishikawa, Neural Networks **9**, 509 (1996).
[8] J. L. Fox, Science **225**, 820 (1984).