2층 다단 신경망회로 코어넷의 처리용량에 관한 연구

박 종 준[†]

요 약

신경망 회로의 해석에서 아직 해결하지 못하는 부분이 은닉층(hidden layer)의 해석이다. 본 논문에서는 신경망 회로의 기본적인 구성회로로써 하나의 입력(p levels)과 하나의 출력(q levels)을 갖는 2-layer Core-Net를 정의하고, 이 Core-Net의 처리 가능 용량(the capacity)은 2차원 무게값 공간(weight space)을 나눌 수 있는 영역의 수로, $a_{p,q}=\frac{q^2}{2}p(p-1)-\frac{q}{2}(3p^2-7p+2)+p^2-3p+2$임을 수학적 귀납법으로 증명하였다. 이 Core-Net로 신경망 회로의 중간층 해석이 가능함을 시뮬레이션 예제를 통하여 보였다.

The Capacity of Core-Net: Multi-Level 2-Layer Neural Networks

Jong-Joon Park[†]

ABSTRACT

One of the unsolved problems in Neural Networks is the interpretation of hidden layers. This paper defines the Core-Net which has an input(p levels) and an output(q levels) with 2-layers as a basic circuit of neural network. I have suggested an equation, $a_{p,q}=\frac{q^2}{2}p(p-1)-\frac{q}{2}(3p^2-7p+2)+p^2-3p+2$, which shows the capacity of the Core-Net and have proved it by using the mathematical induction. It has been also shown that some of the problems with hidden layers can be solved by using the Core-Net and using simulation of an example.

1. Introduction

The major purpose of using neural network is a generalization: to have the outputs of the net approximate target values given inputs that are not in the training set. It has been known that there are three conditions which are typically necessary for a good generalization. The first condition is that the inputs to the network contain sufficient information pertaining to the target, so that there exists a mathematical function relating correct outputs to inputs with the desired degree of accuracy. The second condition is the smoothness; a small change in the inputs should, most of times, produce a small change in the outputs. The third condition is that the training samples should be sufficiently large and should be representative subset of the sets of all cases(population).

† 정 회 원 : 서경대학교 전산정보관리학과 교수

논문접수 : 1998년 12월 11일, 심사완료 : 1999년 7월 19일

In MLPs(MultiLayer Perceptrons) with step/threshold Heaviside activation functions, it is needed only two hidden layers to implement full generality.[1][2] In MLPs with any of a wide variety of continuous nonlinear hidden-layer activation functions, one hidden layer with an arbitrarily large number of units suffices for the "universal approximation" property.[3] But there is no theory yet to tell how many hidden units are needed to approximate any given function. A few books and articles suggest several rules to determine the number of units, but it is impossible to determine a good neural network architecture only from the number of inputs and outputs.[4] It critically depends on the number of training samples, the amount of noise(the unnecessary samples), and the complexity of the function(classification) to learn. A rule to find out the number of needed training samples is to use as many hidden units as the number of weights times 10 in the network. This rule is concerned with only overfitting and is unreliable too. The only thing that we can say is that if the number of training samples is much larger than the number of weights, it is unlikely to get overfitting, but likely to suffer from underfitting. Ordinary RBF(Radial-Basis Function) networks containing only a few hidden units also produce peculiar, bumpy output functions. Normalized RBF networks are better at approximating simple smooth surfaces with a small number of hidden units.[5][6]

The intelligent way to decide the number of hidden units depends on using early stopping or some other form of regularization. Otherwise, simply try many networks with different numbers of hidden units, estimate the generalization error for each one, and choose the network with the minimum estimated generalization error.

This paper defines the Core-net: 2 layered multi-level(p levels) input and output(q levels) neural network. The output node of this Core-Net may work as a hidden unit in a complicated multi layered neural networks, since the output node has values of multi levels. I will suggest a theorem which is related to the capacity of the Core-Net and will prove it by mathematical induction. I will also show a simulation example to prove the results.

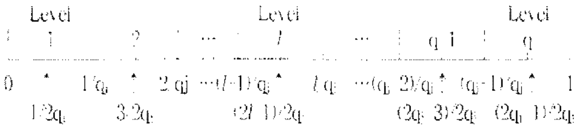
2. Multi-Level 2-Layered Neural Networks

2.1 The Multi-Level Grading Rule (MLGR)

Symbolic values should be converted into numeric values so that a linguistic symbol can be processed in a neural network system. In order to have an equal range for each linguistic value, a MLGR for the conversion of linguistic symbols had been proposed.[7][8] The k_{th} value of the level of a symbol which has L levels in total is represented as

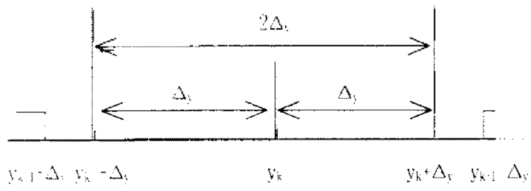
$$\frac{2(k-1)+1}{2L}, \qquad (1)$$

where L is the number of levels in a linguistic expression. Therefore, the maximum effective (decision) range is ±1/2L, i.e., the length of 1/L. In other words, the linguistic effective range must be less than 1/L. Here the effective range means the range for which a linguistic symbol is in effect, that is, the range of the symbol's value as shown in (Fig. 1). The domain of this range is [0,1], because the input values of the neural network and the output of the sigmoidal activation function, Equation 1, lie between 0 and 1.

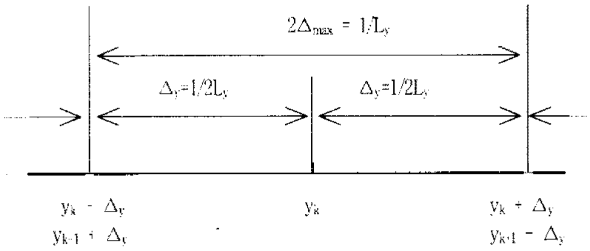


(Fig. 1) Values of each level by grades in qj Levels

Δ_y is defined as half of the effective range of the level(linguistic) term y in the k_{th} level and is shown in (Fig. 2) and (Fig. 3). Here $\varDelta_{max}$ is the maximum effective range.


$2\Delta_y$
Δ_y
Δ_y
$y_{k-1}-\Delta_y$ $y_k-\Delta_y$
y_k
$y_k+\Delta_y$ y_{k+1} Δ_y

(Fig. 2) The effective range Δ_y of a level(linguistic) term y in the k_{th} Level



(Fig. 3) The maximum effective Range Δ_{max} of a level(linguistic) term y_k in the k_{th} Level

2.2 The Definition of the Core-Net

Definition. (Core-Net):

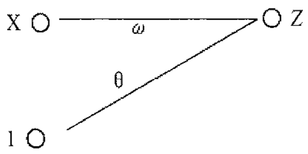
A Core-Net is a two-layered Neural Networks consisting of one input, one threshold and one output node. The input and output nodes may have multi-levels for their values.

The Core-Net is depicted in (Fig. 4) and the multi-leveling is shown in (Fig. 1) and (Fig. 2).

From the sigmoid activation function, we can rewrite the output as,

$$Z = \frac{1}{1+\exp[-(\sum_{i=1}^{n} w_{ji}X_i + \theta_j)]} = \frac{k}{q} \tag{2}$$

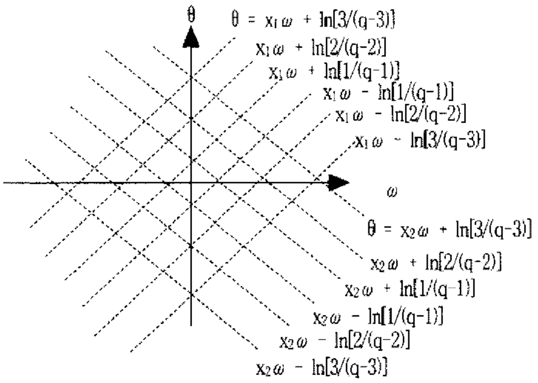
$$\boldsymbol{\theta} = -\boldsymbol{X}\boldsymbol{w} + \ln\frac{k}{q-k} \tag{3}$$



(Fig. 4) One input and one output multi-level neural network

When an input $X = \{x_1, x_2, x_3, \ldots x_p\}$, $|X| = p$, an output $Z = \{z_1, z_2, z_3, \ldots z_q\}$, and the size of Z, $|Z| = q$, the total number of possible function is q^p. The capacity of the neural network of the above equation is the number of segments which are separated by the equations in weight space.

For example, in the case of p=2 and q=7 as shown in (Fig. 5), the number of segments is 49 which is q^p, and the total number of combinatorial function is also 49. Thus, in this case all the functions arc implementable by only a 2-layer neural network.



(Fig. 5) The weight space representation of the weight equations (p=2, q=7)

2.3 The capacity (the number of separable regions) of the Core-Net

The number of implementable function(the capacity) is the subset of the possible functions, for example q^p in one input with p levels and one output with q levels.

Theorem (The Capacity of the Core-Net):

Let the capacity(the number of separable regions in the Core-Net with p input levels and q output levels; there are $p(q-1)$ lines) is $a_{p,q}$, for $p,q \in \boldsymbol{N}$, $p \geq 1$ and $q \geq 2$. Here, $\boldsymbol{N}$ is the set of natural number. Then the capacity $a_{p,q}$, is as follows:

$$a_{p,q} = \frac{q^2}{2}p(p-1) - \frac{q}{2}(3p^2-7p+2) + p^2 - 3p + 2. \quad \text{for p} \geq 1 \text{ and q} \geq 2 \tag{4}$$

Proof (by mathematical induction):

1. For q=2, $a_{p,2} = 2p$ (base step).

1) For p=1, that is, it has one line and separates the space (2-D plane) into two areas. So, $a_{1,2}=2$; the line goes through the original position and cuts the plane into two regions. Therefore the equation $a_{p,q}$ holds.

2) Assume $a_{p,2}$ holds (hypothesis step).

3) For $p+1$, the number of new separate regions by adding one line is 2: the line goes through the original position. So, $a_{p+1,2}$ is $a_{p,2}+2$, and is $2(p+1)=a_{p+1,2}$.

So, $a_{p,2}$ holds for any $p\in \boldsymbol{N}$ (the set of Natural Number) and $p\geq 1$.

2. Assume $a_{p,q}$, holds for some $p\in \boldsymbol{N}$, $p\geq 1$, $q\in \boldsymbol{N}$ and $q\geq 2$ (hypothesis step).

3. Now to prove that $a_{p,q+1}=\frac{qp}{2}(qp-q+5-p)-q+1$ holds for any $p\in \boldsymbol{N}$, $p\geq 1$ and for some $q+1\in \boldsymbol{N}$ and $q\geq 2$, apply the mathematical induction to the equation $a_{p,q+1}$ by the number p.

1) For p=1, there are q parallel lines and it separates the plane as $q+1$ regions. So, $a_{1,q+1}=q+1$, it holds.

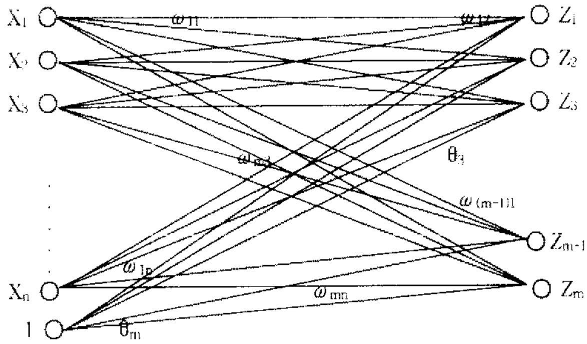
2) Assume $a_{p,q+1}$ holds; it has p sets of q+1 parallel lines(refer the equation 7 and (Fig. 5)) (hypothesis step).

3) For $p+1$, by adding one set of q+1 parallel lines, the additional number of separable region is $q[p(q-1)+2]$. Therefore, $a_{p+1,q+1}$ is $a_{p,q+1}+q(qp-p+2)$ and is the same as $a_{p+1,q+1}$ of the theorem. So, $a_{p,q+1}=\frac{qp}{2}(qp-q+5-p)-q+1$holds for any $p\in \boldsymbol{N}$ and $p\geq 1$.

4. Therefore $a_{p,q}$ holds for any p, $q\in \boldsymbol{N}$, $p\geq 1$, and $q\geq 2$. ■

2.4 2-Layered Multi-Level Neural Network using MLGR(Multi-Level Grading Rule)

The 2-layer multi-level neural network is composed of an input layer and an output layer without hidden layer. Each input has p levels and each output has q levels. This has been depicted in (Fig. 6).



(Fig. 6) Two-layer multi-level neural network

where ω_{ji} is the weight value between the node of X_i and Z_j, θ_j is the threshold value of output node Z_j, $X_i \in \{x_1, x_2, x_3, \ldots x_p\}$, $|X_i| = p$, $Z_j \in \{z_1, z_2, z_3, \ldots z_q\}$, $|Z_j| = q$, for any $1\leq i\leq n$ and $1\leq j\leq m$.

Therefore, the output of Z_j is

$$Z_j=\frac{1}{1+\exp[-(\sum_{i=1}^{n} w_{ji}X_i+\theta_j)]}. \qquad (5)$$

When $|Z_j| = 2$, let $Z_j = 1/2$ then

$$w_{j1}X_1+w_{j2}X_2+\cdots+w_{ji}X_i+\cdots+w_{jn}X_n+\theta_j=0, \quad (6)$$

$$\begin{pmatrix}\theta_1\\ \theta_2\\ \vdots\\ \theta_j\\ \vdots\\ \theta_m\end{pmatrix}=\begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1i} & \cdots & w_{1n}\\ w_{21} & w_{22} & \cdots & w_{2i} & \cdots & w_{2n}\\ \vdots & \vdots & & \vdots & & \vdots\\ w_{j1} & w_{j2} & \cdots & w_{ji} & \cdots & w_{jn}\\ \vdots & \vdots & & \vdots & & \vdots\\ w_{m1} & w_{m2} & \cdots & w_{mi} & \cdots & w_{mn}\end{pmatrix}\begin{pmatrix}X_1\\ X_2\\ \vdots\\ X_i\\ \vdots\\ X_n\end{pmatrix}, \quad (7)$$

or

$$\begin{pmatrix}w_{1i}\\ w_{2i}\\ \vdots\\ w_{ji}\\ \vdots\\ w_{mni}\end{pmatrix}=-\frac{1}{X_i}\begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1i} & \cdots & w_{1(n-1)}\\ w_{21} & w_{22} & \cdots & w_{2i} & \cdots & w_{2(n-1)}\\ \vdots & \vdots & & \vdots & & \vdots\\ w_{j1} & w_{j2} & \cdots & w_{ji} & \cdots & w_{j(n-1)}\\ \vdots & \vdots & & \vdots & & \vdots\\ w_{m1} & w_{m2} & \cdots & w_{mi} & \cdots & w_{m(n-1)}\end{pmatrix}\begin{pmatrix}X_1\\ X_2\\ \vdots\\ X_i\\ \vdots\\ X_{(n-1)}\end{pmatrix}$$

$$-\frac{1}{X_i}\begin{pmatrix}\theta_1\\ \theta_2\\ \vdots\\ \theta_j\\ \vdots\\ \theta_m\end{pmatrix} \quad (8)$$

Let us extend this equation to $|Z_j| = q_j$, $\mathbf{Z}_j$ = $\{1/2q_j, 3/2q_j, \ldots, (2k\ 1)/2q_j, \ldots, (2q_j-1)/2q_j\}$ and Z_{jk}

$= (2k-1)/2q_j$, where $k = 1, 2, \dots q_j$, i.e., Z_{jk} is the k_{th} level of q_j-levels in the j_{th} output by the MLGR(refer the (Fig. 1)). Let us consider the border of each level, i.e., $Z_{jk} = k/q_j$ in q_j levels to find out the weight space separability (the capacity of a neural network), then

$$Z_{jk} = \frac{1}{1+\exp[-(\sum_{i=1}^{n} w_{ji}X_i + \theta_j)]},$$
$$= \frac{k}{q_j} \tag{9}$$

$$\begin{pmatrix}\theta_1\\ \theta_2\\ \vdots\\ \theta_j\\ \vdots\\ \theta_m\end{pmatrix} = -\begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1i} & \cdots & w_{1n}\\ w_{21} & w_{22} & \cdots & w_{2i} & \cdots & w_{2n}\\ \vdots & \vdots & & \vdots & & \vdots\\ w_{j1} & w_{j2} & \cdots & w_{ji} & \cdots & w_{jn}\\ \vdots & \vdots & & \vdots & & \vdots\\ w_{m1} & w_{m2} & \cdots & w_{mi} & \cdots & w_{mn}\end{pmatrix}\begin{pmatrix}X_1\\ X_2\\ \vdots\\ X_i\\ \vdots\\ X_n\end{pmatrix} + \ln\frac{k}{q_j - k}, \tag{10}$$

where $1 \le k \le q_j - 1$.

2.5 The Number of Functions in *p* input levels and *q* output levels

Multi-valued function f with domain $\boldsymbol{X} = \{x_1, x_2, x_3, \dots x_p\}$, $|\boldsymbol{X}| = p$, $x_i = \frac{2i-1}{2p}$, $1 \le i \le p$, and range $Z = \{z_1, z_2, z_3, \dots z_q\}$, $|Z| = q$, $z_j = \frac{2j-1}{2q}$, and $1 \le j \le q$, for some integer p and q, is as follows:

$$f : \{x_1, x_2, x_3, \dots x_p\}^n \rightarrow \{z_1, z_2, z_3, \dots, z_q\},$$
$$f(X_1, X_2, \dots, X_n) = Z. \tag{11}$$

Here, $\mathbf{X}^n$ denotes the n-fold cartesian product of the set $\mathbf{X}$ with itself.

Let the set $\boldsymbol{X}$ is the set of all possible inputs(the combination of n inputs with p levels for each of them) as follows:

$$\boldsymbol{X} = \{(X_1, X_2, \dots, X_i, \dots, X_n) \mid X_i \in \{x_1, x_2, \dots, x_j, \dots, x_p\},$$

$$\text{where } x_j = \frac{2j-1}{2p}, 1 \le j \le p, 1 \le i \le n\} \tag{12}$$

$$|\boldsymbol{X}| = p^n. \tag{13}$$

The set Z is the set of all possible outputs(the combination of q levels output for each p^n inputs).

$$\boldsymbol{Z} = \{(Z_1, Z_2, \dots, Z_i, \dots, Z_{p^n}) \mid Z_i \in Z\}, |\boldsymbol{Z}| = q^{p^n},$$
$$\boldsymbol{F} = \{f \mid f: \boldsymbol{X} \rightarrow \boldsymbol{Z}\}, \tag{14}$$

$$|\boldsymbol{F}| = q^{p^n}. \tag{15}$$

If the system has m outputs, then the function of n inputs and m outputs is as follows:

$$\mathfrak{I} : X^n \rightarrow Z^m,$$
$$\text{where } X = \{x_1, x_2, \dots, x_p\},\ Z = \{z_1, z_2, \dots, z_q\}$$
$$0 < x_i < 1, \text{ and } 0 < z_i < 1. \tag{16}$$

$$\mathfrak{I} = \{\mathfrak{I} \mid \mathfrak{I} : \boldsymbol{X} \rightarrow Z^m\}. \tag{17}$$

The total number of possible function is

$$|\mathfrak{I}| = (q^{p^n})^m = q^{mp^n}. \tag{18}$$

So, the total number of functions by combination of p levels in n inputs and q levels in m outputs is q^{mp^n}.

A Boolean function is a function f with domain {F,T} and range {F,T}, for some integer n and m as follows:

$$f : \{F, T\}^n \rightarrow \{F, T\}^m, \tag{19}$$

where $\{F,T\}^n$ denotes the n-fold cartesian product of the set $\{F,T\}$ with itself. When a problem has n input variables and m output variables, then there are $|f| = 2^{m2^n}$ kinds of functions (rules) by their combination of inputs and outputs. <Table 1> shows

〈Table 1〉 The possible combinatorial functions(rules) for 2 inputs with binary levels of inputs and output

	Inputs		The Functions of Outputs Z															
No.	X	Y	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}
1	F	F	F	F	F	F	F	F	F	F	T	T	T	T	T	T	T	T
2	F	T	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T	T
3	T	F	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T
4	T	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T

an example which has two inputs with binary levels and one output with binary levels, and it has $16(= 2^{2^2})$ kinds of possible functions(rules).

2.6 The Separability in a d-Dimensional Hyperspace in General Position

There are C(N, d) homogeneously linearly separable dichotomies of N points in general position in Euclidean d-space, where

$$C(N,d) = 2\sum_{k=0}^{d-1}\binom{N-1}{k}. \qquad (20)$$

The above equation is proved by mathematical induction.[6][11] For all real s and integer k, the binomial coefficients comprising (N, d) is defined by

$$\binom{s}{k} = \frac{s(s-1)\cdots(s-k+1)}{k!}.$$

Therefore the maximum capacity of a multi-layer neural network with N hidden nodes and d-dimension of input space is C(N, d). Actually the capacity of this neural network is far less than C(N, d). The capacity of a simple binary level perceptron of an associative network of N nodes has known as 0.138N.[10]

3. The Simulation of an Example

3.1 Model 1(3)-1(2)

The model expression "1(3)-1(2)" means that the Core-Net has one input with 3 levels and 1 output with 2 levels. In this case, the number of all possible functions with p=3 and q=2 is 8 (= 2^3). Substitute p and q into the equation (4), then $a_{3,2}$ is 6. There are 6 regions, which are separable with 3 lines derived from the equation (10), as shown in (Fig. 7). Thus, there are two functions which are not implementable with this 1(3)-1(2) neural network out of 8 combinatorial possible functions. The simulation results are shown on section 3.2.

3.2 The Simulation Results of model 1(3)-1(2)

The model has been tested with an input and an output which have multi-levels with backpropagation algorithm in neural network. The effective ranges of all the level values had been set to 10% of the maximum range Δ_{max}; the learning rate eta was 0.9; the momentum factor alpha was 0.7; the maximum iteration number was 32767. The system was run 10 times with variable random initial weight value sets in each 8 possible input combinations (functions). For example, in the case of function 1(F_2), i.e., output of F, F, T, the training (input, output) pairs were (0.17, 0.25), (0.50, 0.25), and (0.83, 0.75) and the generated output results were (0.17, 0.128825), (0.50, 0.363400), and (0.83, 0.687856). The simulation results of 1(3)-1(2) model are shown in <Table 2>. The first column of <Table 2> is the function number. The second one is the test number with the number of iteration. The third one is the generated output. The fourth one is the number of correct outputs out of three data(levels). The fifth one is the weight value. The sixth one is the theta(bias). Finally, the seventh one is the generalized error.

In <Table 2>, the simulation of functions F_2 through F_7 were also run 10 times each, but they were not converged to the 10% of effective range of error and reached the maximum iteration. Thus, the generated outputs, ω's, Theta's, and Err's are same in each run. In the results of functions 3 and 6 (F_3, F_6), the number of correct outputs were only 2 out of 3 samples in each other. This means that the two functions are not implementable with this 1(3)-1(2) neural network system and the Errs are big. Another run of this system with the effective range of 100% and with the number of maximum iteration of 1,000,000 shows that only the simulation of these two functions (F_3, F_6) are reached the maximum iteration number. This confirms that these functions are not implementable with this 1(3)-1(2) model.

If the effective range is set wide, the number of iteration is small to converge and the weight values are located near the separate lines (hyper-planes in multi-dimensional weight space) but inside of them.

〈Table 2〉 Simulation results of model 1(3)-1(2)

Func. #	Iter.	Generated Outputs			# of Corr.	ω	Theta	Err ($\times 10^{3}$)
		Sample 1	Sample 2	Sample 3				
F_1	1. 33	0.274809	0.250990	0.228586	3	-0.368967	-0.908853	0.189628
	2. 3	0.270739	0.271544	0.272351	3	0.012226	-0.992914	0.962583
	3. 5	0.271060	0.250243	0.230520	3	-0.324220	-0.935206	0.183204
	4. 4	0.268320	0.246312	0.225552	3	-0.345690	-0.945534	0.42889
	5. 16	0.274873	0.251050	0.228640	3	-0.368992	-0.908527	0.189869
	6. 4	0.263372	0.252684	0.242287	3	-0.167514	-1.000591	0.200872
	7. 4	0.258336	0.247265	0.236517	3	-0.175870	-1.025320	0.208444
	8. 4	0.264380	0.246421	0.229302	3	-0.283422	0.976081	0.337266
	9. 5	0.271981	0.250269	0.229744	3	-0.337773	-0.928290	0.197172
	10. 22	0.274839	0.250989	0.228557	3	-0.369443	-0.908621	0.190123
F_2	32767	0.128825	0.363400	0.687856	3	4.052664	-2.586968	5.39767
F_3	32767	0.420079	0.418220	0.416362	2	-0.022917	-0.318629	29.3043
F_4	32767	0.313221	0.636600	0.870612	3	4.037639	-1.458180	5.39768
F_5	32767	0.686779	0.363399	0.129388	3	-4.037638	1.458179	5.39768
F_6	32767	0.579921	0.581780	0.583638	2	0.022916	0.318629	29.3043
F_7	32767	0.871175	0.636600	0.312144	3	-4.052664	2.586968	5.39768
F_8	1. 4	0.736800	0.744835	0.752708	3	0.125573	1.008467	0.1614
	2. 41	0.725385	0.749005	0.771233	3	0.365965	0.910328	0.186578
	3. 15	0.725210	0.748873	0.771142	3	0.366507	0.909358	0.187346
	4. 13	0.725049	0.749032	0.771583	3	0.371475	0.907721	0.191296
	5. 4	0.729446	0.749474	0.768490	3	0.312019	0.939798	0.360193
	6. 30	0.725350	0.749005	0.771266	3	0.366511	0.910060	0.18713
	7. 30	0.725301	0.749007	0.771312	3	0.367267	0.909688	0.187898
	8. 20	0.725174	0.749006	0.771423	3	0.369173	0.908734	0.189873
	9. 11	0.725151	0.750073	0.773442	3	0.386572	0.905717	0.199091
	10. 4	0.733050	0.749251	0.764785	3	0.253442	0.967903	0.286947

When the effective range is set narrow (f.g., 10%), the trained weight values are located near the center of the regions. (Fig. 7) graphically shows the drawing of the execution results of the first runs in each of 8 functions, as shown in the <Table 2>.

(Fig. 7) shows that there is only one point in each regions except the point of p3 and p6, which are the outputs of the run those are not converged and the results are not correct.

$\theta 1 = -1/6\ \omega$

$\theta_2 = -3/6\ \omega$

$\theta_3 = -5/6\ \omega$

p1=(-0.368967, -0.908853)

p2=(4.052664, -2.586968)

p3=(-0.022917, -0.318629)

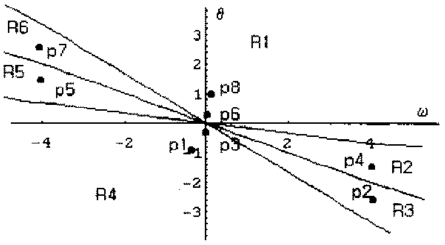
p4=(4.037639, -1.458180)

p5=(-4.037638, 1.458179)

p6=(0.022916, 0.318629)

p7=(-4.052664, 2.586968)

p8=(0.125573, 1.008467)



(Fig. 7) 1(3)-1(2), one input with 3 levels and 1 output with 2 levels

4. Conclusion

This paper suggests a theorem about the equation to find out the capacity(number of separable regions) of the Core-Net: two layered multilevel neural network (refer to the equation 4). The equation has been applied to an example of model 1(3)-1(2): one input with 3 levels and 1 output with two levels. This model has been run with an artificial neural network using backpropagation algorithm and the results are shown in <Table 2>.

The number of combinatorial functions in this model is 8 as described in section 2.3 and in section 3.1. The graph in (Fig. 7) shows that there are 6 separable regions(R1 through R6 in clockwise). Therefore, there are two functions which are not implementable. These points p3 and p6, which are not converged, are located in wrong positions in the graph and their outputs are incorrect as shown in <Table 2>. From this output results, the unimplementable (reached the maximum iteration) functions are exactly matched to (Fig. 7) as described in section 2.3, and to the theorem $a_{p,q}$.

The weight values of the simulation are located inside the separable regions as shown in the figure. On the other hand, the weight points of the two un-implementable functions are located in the wrong place and have no meaning.

This Core-Net could be applied to the multi-layered neural network. The Core-Net is very helpful to solve an optimized neural network problems such as optimum nodes, links, and the optimum number of training data. The input and output levels can be interpreted as linguistic symbols for the process of approximate reasoning too.

References

[1] Sontag, E.D. "Feedback stabilization using two-hidden-layer nets," IEEE Transactions on Neural Networks, 3, pp.981-990, 1992.

[2] Bishop, C.M. "Neural Networks for Pattern Recognition," Oxford: Oxford University Press, pp. 121-126, 1995.

[3] Ripley, B.D. "Pattern Recognition and Neural Networks," Cambridge:Cambridge University Press, pp.173-180, 1996.

[4] Weigend, A. "On overfitting and the effective number of hidden units," Proceedings of the 1993 Connectionist Models Summer School, pp.335-342, 1994.

[5] Hirose, Yoshio, Yamashita, Koichi, and Hijiya, Shimpei, Back-Propagation Algorithm Which Varies the Number of Hidden Units, Neural Networks, Vol.4, pp.61-66, 1991.

[6] Thomas M. Cover, "Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition" by IEEE Transactions on Electronic Computers, 1965, pp.326-334.

[7] Park, Jong Joon, Neural Network Processing of Linguistic Symbols Using Multi-Level Grading Rule, Ph.D. Dissertation, The Florida State University, Tallahassee, Florida, USA, 1993.

[8] Park, Jong Joon, Abraham Kandel, G. Langholz and L. Hawkes, Neural Network Processing of Linguistic Symbols, Fuzzy Sets, Neural Networks, and Soft Computing, Ed. by R.R. Yager and L.A.Zadeh, Van Nostrand Reinhold, pp.265-284, 1994.

[9] Shoemaker, Patrick A., Carlin, Michael J., and Shimabukuro, Randy L., Back Propagation Learning With Trinary Quantization of Weight Updates, Neural Networks, Vol.4, pp.231-141, 1991.

[10] John Hertz, Anders Krogh, and Richard G. Palmer, Introduction to the Theory of Neural Computation, Addison-Wesley, 1991.

[11] Gagan Mirchandani and Wei Cao, On Hidden Nodes for Neural Nets, IEEE Transactions on Circuits and Systems, Vol.36, No.5, May 1986, pp.661-664.

[12] McClelland, James L. and Rumelhart, David E., An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An Account of Basic Findings, Psychological Review, Vol.88, No.5, pp.375-407, September 1981.

박 종 준

e-mail : jong@bukak.seokyeong.ac.kr

1978년 서강대학교 물리학과(학사)

1980년 연세대학교 전자공학과(공학석사)

1994년 미국 the Florida State University 전산학과(Ph.D)

1995년~현재 서경대학교 전산정보관리학과 조교수

관심분야 : 인공지능, 신경망회로, 퍼지이론, 패턴인식, 정보처리