논문 2004-41CI-3-7

다층 신경회로망을 위한 자기 구성 알고리즘

(A self-organizing algorithm for multi-layer neural networks)

이 종 석*, 김 재 영**, 정 승 범***, 박 철 훈****

(Jong-Seok Lee, Jae-Young Kim, Seung-Beom Chung, and Cheol Hoon Park)

요 약

신경회로망을 이용하여 주어진 문제를 해결할 때, 문제의 복잡도에 맞는 구조를 찾는 것이 중요하다. 이것은 신경회로망의 복잡도가 학습능력과 일반화 성능에 크게 영향을 주기 때문이다. 그러므로, 문제에 적합한 신경회로망의 구조를 자기 구성적으로 찾는 알고리즘이 유용하다. 본 논문에서는 시그모이드 활성함수를 가지는 전방향 다층 신경회로망에 대하여 주어진 문제에 맞는 구조를 결정하는 알고리즘을 제안한다. 개발된 알고리즘은 구조증가 알고리즘과 연결소거 알고리즘을 이용하여, 주어진 학습 데이터에 대해 가능한 한 작은 구조를 가지며 일반화 성능이 좋은 최적에 가까운 신경회로망을 찾는다. 네 가지 함수 근사화 문제에 적용하여 알고리즘의 성능을 알아본다. 실험 결과에서, 제안한 알고리즘이 기존의 알고리즘 및 고정구조를 갖는 신경회로망과 비교하였을 때 최적 구조에 가까운 신경회로망을 구성하는 것을 확인한다.

Abstract

When a neural network is used to solve a given problem, it is necessary to match the complexity of the network to that of the problem because the complexity of the network significantly affects its learning capability and generalization performance. Thus, it is desirable to have an algorithm that can find appropriate network structures in a self-organizing way. This paper proposes algorithms which automatically organize feedforward multi-layer neural networks with sigmoid hidden neurons for given problems. Using both constructive procedures and pruning procedures, the proposed algorithms try to find the near optimal network, which is compact and shows good generalization performance. The performances of the proposed algorithms are tested on four function regression problems. The results demonstrate that our algorithms successfully generate near-optimal networks in comparison with the previous method and the neural networks of fixed topology.

Keywords: self-organizing algorithm, multi-layer neural networks, construction, pruning, impact factor

I. 서 론

인공신경회로망(artificial neural network)은 인간의 두뇌를 모방하여 여러 실세계의 문제를 푸는데 응용되어왔다. 대규모 병렬분산구조를 이용하여 경험적인 지식을 예제에 의한 학습을 통해 일반화할 수 있는 능력이 있기 때문에[1] 최적화, 음성인식, 패턴인식, 적응제어, 신호처리 등의 분야에서 널리 적용되고 있다[2].

신경회로망을 어떤 문제에 적용할 때 가장 먼저 겪게 되는 문제 중 한 가지는 적절한 구조를 결정하는 것이다. 일반적으로, 주어진 문제의 복잡도에 비해 신경회로망의 구조가 너무 작으면 문제를 충분히 학습하지 못하게 되고, 너무 복잡하면 학습이 필요이상으로 되어 일반화 성능이 나빠지는 과도학습(overfitting)현상이 발생한다[1]. 따라서, 문제에 적합한 구조, 즉 학습이 가능하

* 학생회원, **** 정회원, 한국과학기술원 전자전산학과
(Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology)
** 정회원, 서두인칩
(Seodu Inchip co., LTD)
*** 정회원, 삼성전자(주) 기술총괄 소프트웨어센터
(Samsung Electronics, Software Center)
※ 본 연구는 한국과학재단 목적기초연구 (R01-2003-000-10829-0) 지원으로 수행되었습니다.
접수일자: 2003년10월15일, 수정완료일: 2004년4월14일

며 좋은 일반화 성능을 보여주는 작은 크기의 신경회로망을 결정하는 것은 매우 중요한 문제이다.

이를 해결하기 위해서 신경회로망의 구조를 구성할 수 있는 여러 가지 알고리즘이 제시되었는데 다음과 같이 크게 두 가지로 나눌 수 있다[3].

- 구조증가 알고리즘 (construction algorithm)
- 연결소거 알고리즘 (pruning algorithm)

구조증가 알고리즘은 작은 구조에서부터 학습을 시작해서 은닉뉴런과 층의 개수를 증가시켜 가는 방법이다. 연결소거 알고리즘은 이와는 반대로 학습 가능한 충분히 큰 구조로 시작하여 일부 필요 없는 뉴런이나 가중치를 소거하여 신경회로망의 크기를 줄여가는 방법이다. 연결소거 알고리즘은 어느 정도 문제의 복잡도에 대한 사전정보에 근거하여 초기 구조를 정해야 하는 반면, 구조증가 알고리즘은 사전에 문제의 복잡도(com-plexity)에 대해 알지 못해도 사용 가능하다는 장점이 있다. 한편, 구조증가 알고리즘의 단점으로는 크기를 늘려가면서 생성된 신경회로망에는 불필요한 뉴런이 존재할 가능성이 있다는 것이다. 따라서, 본 논문에서는 이 두 알고리즘을 모두 사용하여 각각의 장단점을 보완한 알고리즘을 제안한다.

제안되는 알고리즘은 일반적으로 널리 쓰이는 구조증가 알고리즘의 하나인 동적노드생성 (DNC: dynamic node creation)[4,5] 방법에 기반을 두고 있다. DNC는 비교적 간단한 과정의 알고리즘으로서, 시그모이드 특성의 은닉뉴런을 추가하고 전체 신경회로망을 학습하는 과정을 반복하여 문제에 적합한 신경회로망을 구성한다. 이 방식은 간단하고 수렴성이 보장되는 장점이 있으며, 여러 가지 문제들에서 성공적으로 응용되어 왔다[3]. 본 논문에서 제안하는 알고리즘은 DNC의 기본적인 과정 외에, 첨가되는 뉴런의 초기 가중치(initial weight)를 최적화하는 방법을 사용하며, 뉴런의 추가와 더불어 은닉층의 증가 및 뉴런 제거과정을 포함함으로써 그 성능을 향상시키고자 한다. 알고리즘의 중요한 목표는 주어진 문제에 대하여 *좋은 일반화 특성*을 가지면서 *학습이 가능한 가장 작은* 신경회로망을 *신뢰성 있게* 찾는 것이다. 여기서 신뢰성이란, 신경회로망 학습에서 사용되는 여러 가지 다른 난수(random value)에 의해 결정되는 초기 가중치에 따른 최종 신경회로망의 구조와 일반화 성능이 가능한 한 일정하게 유지되는 것을 의미한다. 이것은 사용자의 측면에서 중요한 문제로서 여러 번의 신경회로망 구성을 수행하였을 때 그 결과가 매번 크게 차이가 난다면 바람직하지 않기 때문이다.

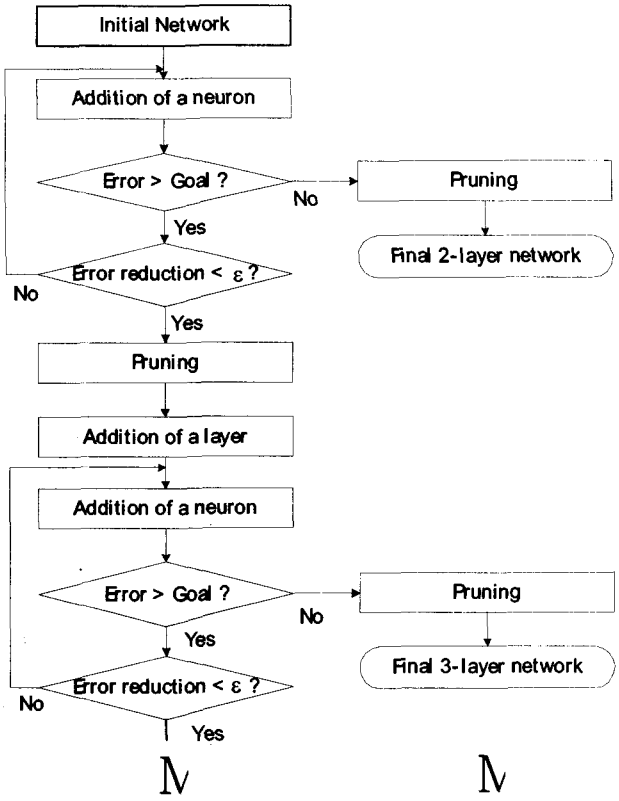


그림 1. 자기 구성 알고리즘의 흐름도
Fig. 1. A flow chart of the overall self-organizing algorithm.

모의실험에서는 제안된 알고리즘을 잘 알려진 네 가지의 함수 근사화 문제에 적용하여 그 성능을 알아본다. 그 결과를 DNC 알고리즘 및 고정된 구조를 가지는 신경회로망과 비교한다.

II. 자기 구성 알고리즘

제안된 알고리즘은 구조증가 과정과 연결소거 과정을 모두 이용한다. 즉, 작은 크기의 신경회로망에서 시작하여 뉴런과 층을 증가시키고, 최종적으로 불필요한 뉴런의 제거과정을 거쳐 신경회로망의 구조를 최적화한다.

그림 1은 자기구성 알고리즘의 전체 과정을 나타낸다. 먼저, 하나의 은닉층을 가지는 초기 네트워크(initial network)가 선택되고, 주어진 오차목표(error goal)에 도달할 때까지 새로운 뉴런이 신경회로망에 추가된다. 각 뉴런이 추가된 후에는 전체 신경회로망을 레벤버그-

마퀴트(LM: Levenberg-Marquardt) 알고리즘[6]에 의해 학습한다. 뉴런의 첨가로 인해 학습오차(training error)가 개선되는 정도가 정해진 비율(ϵ)보다 적고 학습목표치에 도달하지 못했을 때에는 일부 중요하지 않은 은닉뉴런을 제거한 후 새로운 은닉층을 추가한다. 그리고 나서 학습오차가 목표치에 도달할 때까지 새로운 은닉층에 뉴런이 첨가되는 과정이 반복된다. 마지막으로 학습목표를 달성한 후에는 불필요한 은닉뉴런을 제거한다. 이상의 과정은 셋 이상의 은닉층을 가지는 신경회로망을 생성하도록 계속될 수 있다.

1. 초기 네트워크

구조증가 과정은 초기 네트워크의 선택으로 시작한다. 초기 네트워크는 하나의 은닉층과 그 안에 한 개의 은닉뉴런을 갖는다.

초기 네트워크는 신경회로망 구성과정 전체에 영향을 미치게 된다. 서로 다른 초기 가중치를 갖는 초기 네트워크는 각각 다른 구조와 일반화 성능을 가지는 신경회로망으로 성장하게 된다. 그러므로 여기서는 가능한 한 최적의 초기 네트워크를 찾는 방법을 제안한다.

입력에서 은닉층으로의 연결가중치는 가중치 공간(weight space)에서 균일하게 분포된 가중치벡터 중에서 가장 좋은 것을 선택한다. 신경회로망의 입력이 2차원일 때, 가중치 공간의 극좌표계에서 각도 성분의 $[0,\pi]$ 구간을 N_C개로 균등하게 나누고* 일정한 길이를 부여하여 후보 벡터들을 생성한다. 즉, 입력가중치(input weight) 벡터 $\mathbf{w} = [w_1, w_2]$의 N_C개의 후보는 다음과 같이 생성된다.

$$w_1 = r\cos\left(\frac{\pi i}{N_c}\right)$$
$$w_2 = r\sin\left(\frac{\pi i}{N_c}\right) \quad (1)$$

여기서 $i = 0, 1, ..., N_c - 1$이고, r은 가중치 벡터의 길이이다. r값으로는 작은 값을 선택하는 것이 학습을 용이하게 하고 과도학습 현상을 막을 수 있다. 본 논문의 실험에서는 이 값을 0.05로 하였다.

각 후보 뉴런의 입력바이어스(bias)는 시그모이드 함수의 결정경계(decision boundary)가 학습목표(training target)가 최대가 되는 지점을 지나도록 결정한다. 이는 새로운 뉴런이 학습오차를 효과적으로 줄이도록 함과 동시에 시그모이드 함수가 입력 도메인 내부에서 동작하도록 하기 위한 것이다. 학습샘플 $\mathbf{x}_e$에 대한 학습목표가 최대라고 할 때, $\mathbf{x}_e$를 지나는 후보 뉴런의 결정경계의 방정식은 다음과 같이 쓸 수 있다.

$$\mathbf{w}^1 \cdot \mathbf{x}_e + b^1 = 0 \quad (2)$$

여기서, $\mathbf{w}^1$은 뉴런의 입력가중치, b^1은 입력 바이어스이다. 그러므로 입력 바이어스는,

$$b^1 = -\mathbf{w}^1 \cdot \mathbf{x}_e \quad (3)$$

로 주어진다.

출력 가중치는 출력뉴런이 선형함수를 가지므로 학습 데이터에 대해 유사역원(pseudo-inverse) 방법으로 결정된다.

이렇게 생성된 N_C개의 후보들 중에서 학습오차가 가장 작은 것을 초기 네트워크로 선택한다.

2. 뉴런의 첨가

새로운 뉴런이 추가될 때, 뉴런의 시그모이드 함수의 초평면(hyperplane)의 초기 기하학적 위치는 학습 과정에 영향을 미친다. 그러므로 좋은 성능의 신경회로망을 생성하기 위해서는 새로운 뉴런의 초기 가중치를 최적화하는 방법이 필요하다.

그림 2는 새로운 은닉뉴런을 추가하는 것을 나타낸 그림이다. 새로운 뉴런과 이웃한 층들을 잇는 가중치를

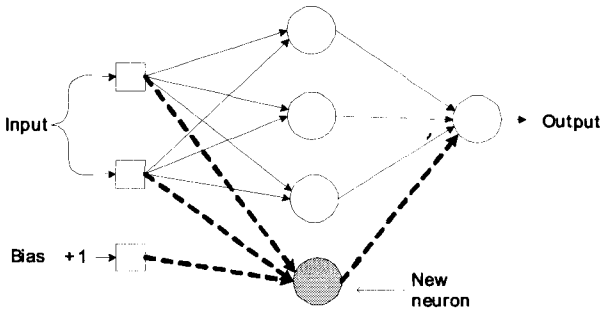


그림 2. 새로운 은닉뉴런의 추가. 점선은 새로운 뉴런과 이웃한 층을 잇는 새로운 가중치를 나타낸다.
Fig. 2. Addition of a new hidden neuron. The dotted lines denote the new weights connecting the new neuron and the existing neurons in the adjacent layers.

* $[0,\pi]$의 범위만 생각한다. $[\pi, 2\pi]$에 위치한 벡터는 $[0,\pi]$에 위치한 벡터에 반대 부호를 취하여 얻을 수 있다. 반대 부호는 출력층에서 고려될 수 있다.

초기화하는 방법으로 두 가지를 제안한다.

첫 번째 방법은 후보집합(pool of candidates)을 사용하는 방법이다. 후보집합은 난수발생으로 생성된 몇 개의 서로 다른 가중치를 가지는 뉴런으로 이루어져 있으며, 그 중에서 현재까지 구성된 신경회로망의 남은 오차(residual error)를 최소화할 수 있는 것이 선택된다. 각 후보 뉴런의 입력 바이어스와 출력 가중치는 초기 네트워크에서와 마찬가지로 정해진다. 즉, 바이어스는 식 (3)에서와 같이 결정경계가 학습오차가 최대가 되는 지점을 지나도록 하며, 출력가중치는 유사역원 방법으로 구한다. 후보들 중에서 가장 성능이 좋은 것을 선택하여 추가한 뒤, LM 알고리즘에 의해 전체 신경회로망을 학습시킨다.

두 번째 방법은 은닉뉴런들의 시그모이드 함수의 결정경계의 기울기 분포를 가능한 한 균일하게 하는 방법이다. 먼저, 신경회로망의 모든 결정경계의 구좌표계(spherical coordinates)에서의 각 위치(angular posi-tion)를 계산한다. 그리고 각각의 좌표축 별로 두 인접한 결정경계 사이가 가장 넓은 영역의 한가운데를 선택하여 새로운 입력 가중치를 위치시킨다. 그림 3은 이러한 과정의 예를 보여준다. 기존의 세 개의 은닉뉴런에 대한 결정경계 h_1, h_2, h_3를 모두 원점을 지나도록 평행이동하여 입력 도메인에 나타내었다. h_1과 h_2 사이의 각이 최대이므로 그 중앙에 새로운 은닉뉴런에 대한 결정경계 h_{new}가 위치하게 된다.

여기서, 가장 넓은 영역의 중앙에 새로운 은닉뉴런의 결정경계를 위치하도록 하는 것 대신, 모든 인접한 두 결정경계 사이에 가능한 새로운 결정경계들을 생성하고 이들 중에서 가장 좋은 것을 선택하는 방법도 가능하다. 그러나 실험결과 후자의 성능이 떨어지는 것으로 나타났다. 이것은 결정경계를 여러 후보 중에서 선택하는 것이 현재에는 가장 좋은 선택이지만 최종적으로도 항상 좋은 결과를 내는 것을 보장하지는 못하는 것을 의미한다. 따라서, 결정경계를 균일하게 분포시키는 것이 좋은 성능의 신경회로망을 구성하는데 더 유용하다는 것을 알 수 있었다.

두 번째 방법에서의 바이어스와 출력 가중치 역시 초기 네트워크에서와 같은 방법으로 결정된다. 입력 가중치와 바이어스가 결정되었을 때, 같은 결정경계를 가지는 두 가지의 경우가 가능하다. 즉,

$$\begin{aligned} h &: \quad \mathbf{w}^1 \cdot \mathbf{x} + b^1 = 0 \\ h' &: -\mathbf{w}^1 \cdot \mathbf{x} - b^1 = 0 \end{aligned} \tag{4}$$

이 두 경우 중에서 학습오차가 작은 것을 선택한다.

두 번째 방법은 첨가되는 뉴런의 결정에서 난수발생을 필요로 하지 않으므로 주어진 학습데이터에 대하여 항상 같은 결과를 주는 결정적(deterministic) 방법이다.

지금까지 설명한 뉴런 선택 방법에 따라 두 가지 자기 구성 알고리즘이 가능하다. 첫 번째 방법은 SOA-P (self-organizing algorithm using a pool), 두 번째 방법은 SOA-D (deterministic self-organizing algorithm)라 부른다.

3. 뉴런의 제거

주어진 학습 데이터에 대해 정해진 오차범위 이내로 학습이 가능하도록 신경회로망의 크기를 키웠을 경우, 중요도가 상대적으로 낮고 과도학습 상태를 유발하는 뉴런이 존재할 수 있다. 연결소거 과정에서는 이러한 뉴런을 제거하여 일반화 성능을 향상시키는 동시에 신경회로망의 크기를 효과적으로 줄인다.

뉴런의 중요도를 측정하는 방법으로서, 임팩트 팩터(ImF: impact factor)를 사용한다[7]. i 번째 뉴런의 임팩

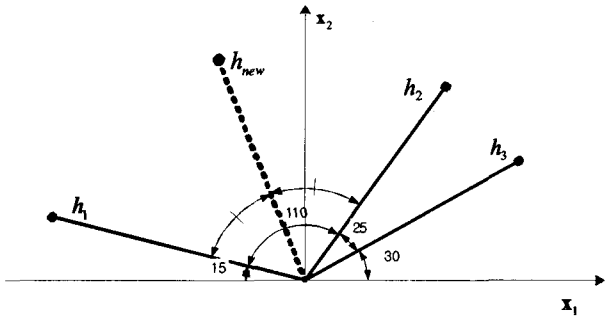


그림 3. 뉴런을 추가하는 두 번째 방법
Fig. 3. Illustration of the second method to add a neuron.

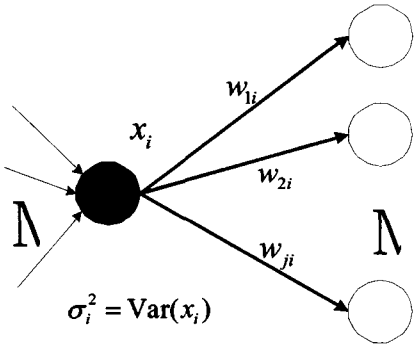


그림 4. 뉴런의 임팩트 팩터의 계산과정
Fig. 4. Calculating the impact factor of a neuron.

트 팩터는 다음과 같이 정의된다.

$$\mathrm{ImF} = \sum_j w_{ji}^2 \sigma_i^2 \tag{5}$$

여기서 σ_i^2는 i 번째 뉴런 출력의 분산이며, w_{ji}는 i 번째 뉴런에서 다음 층의 j번째 뉴런으로의 연결가중치이다 (그림 4). ImF가 작다는 것은 이전층으로부터의 입력이 작거나, 출력의 분산이 작거나, 혹은 다음 층으로의 가중치가 작아서 바이어스와 같은 역할을 하기 때문에 그 뉴런이 학습 오차를 줄이는데 기여하는 바가 적다는 의미이다. 따라서, ImF가 작은 뉴런은 제거될 수 있다.

은닉층이 하나일 경우, 은닉층에서 ImF가 가장 작은 것을 선택하여 제거하고 전체 신경회로망을 재학습시킨다. 만일 오차가 목표만큼 떨어지면 이 과정을 계속 반복하고, 학습에 실패하면 뉴런을 제거하지 않고 제거과정을 끝마친다. 은닉층이 2개 이상일 경우 연결소거는 각 층별로 진행된다. 현재의 은닉층의 뉴런 중에서 가장 작은 ImF를 가지는 것을 제거한 후 전체 신경회로망을 재학습시키는 과정을 모든 은닉층을 번갈아 가면서 반복한다. 모든 층에서의 뉴런제거가 불가능할 때까지 이 과정이 계속된다.

뉴런이 하나 제거될 때, 신경회로망의 학습오차가 급격히 증가하는 것을 방지함과 동시에 신경회로망의 학습이 용이하게 하기 위해, 제거되는 뉴런의 출력의 평균값을 다음 층으로의 바이어스에 더해준다.

뉴런의 소거는 새로운 층을 증가시키기 전과, 구조증가에 의해 학습오차 목표를 만족시키는 신경회로망을 완성한 후 수행한다.

4. 층의 증가

하나의 은닉층을 가지는 신경회로망이 임의의 함수를 근사화할 수 있음이 증명되어 있으나[8], 더 많은 수의 은닉층을 사용함으로써 신경회로망의 크기를 줄일 수 있다[9].

뉴런이 하나씩 증가하면서 오차가 감소하는 양은 점차 줄어드는 경향이 있다. 오차감소가 작아지면 은닉뉴런을 추가하는 것보다는 새로운 층을 더하는 것이 효과적이다. 따라서, 다음의 조건이 만족될 때 은닉뉴런의 추가를 멈추고 층을 하나 증가시킨다.

$$\frac{E(k) - E(k+n)}{E(k)} < \epsilon \tag{6}$$

여기서 $E(k)$는 k번째 뉴런이 추가되어 재학습된 후 신경회로망의 학습오차이다. 즉, k개의 뉴런이 존재하는 신경회로망에 n개의 뉴런을 새로이 추가함으로써 감소된 학습오차의 양과 추가하기 전의 오차의 비가 ϵ 보다 작을 경우 층을 추가한다. 본 논문의 실험에서는 비교간격인 n을 3으로 하였다.

간결한 구조의 신경회로망을 구성하기 위해 층이 추가되기 전에 일부 불필요한 은닉뉴런을 제거한다*. 뉴런을 제거한 후 새로운 은닉층이 추가되며, 추가된 은닉층에 새로운 은닉뉴런들이 추가된다 (그림 5). 층이 추가될 때에는, 먼저 선형함수의 특성을 가지는 현재의 출력뉴런이 새로운 은닉층의 첫 번째 시그모이드 뉴런이 된다. 그리고 나서, 새로운 은닉층에 두 번째 뉴런이 추가되고 새로운 출력뉴런을 추가한다. 기존의 출력뉴런이 선형함수에서 시그모이드 함수로 바뀔 때 학습오차가 급격히 증가하는 것을 막기 위해 기존의 은닉층에서의 가중치(그림 5에서 $\mathbf{w}_a$)를 일정 비율로 스케일링(scaling)을 하여 이 뉴런이 시그모이드의 선형범위 안에서 동작하도록 한다. 그리고 이를 보상하기 위해 이 뉴런으로부터 새로운 출력뉴런으로의 가중치(그림 5에서 $\mathbf{w}_b$)를 역으로 스케일링한다.

III. 실 험

제안된 알고리즘을 함수 근사화 문제에 적용하여 성능을 알아본다. 근사화에 사용된 함수는 2차원의 입력과 하나의 출력을 가지는 함수 네 개를 사용하였다[10-12].

$$g_i : [0,1]^2 \rightarrow \mathbb{R},\ i = 1,2,3,4 \tag{7}$$

함수들은 $[0,1]^2$에서 10,000개의 점을 균일하게 뽑은 입력에 대한 함수값의 표준편차가 1이 되도록 크기가 조정되었으며, 함수값이 0보다 큰 값을 갖도록 이동되었다. 이것은 각 함수간의 성능 비교를 하기 위해서이다. 그림 6은 함수들의 3차원 그림을 나타낸다.

* 이 과정을 생략하는 경우 최종적으로 구성된 신경회로망의 크기가 커지는 것을 관측하였다. 이는 불필요한 은닉뉴런이 제거된 간결한 신경회로망의 경우가 학습 및 연결소거하기 쉽기 때문이다.

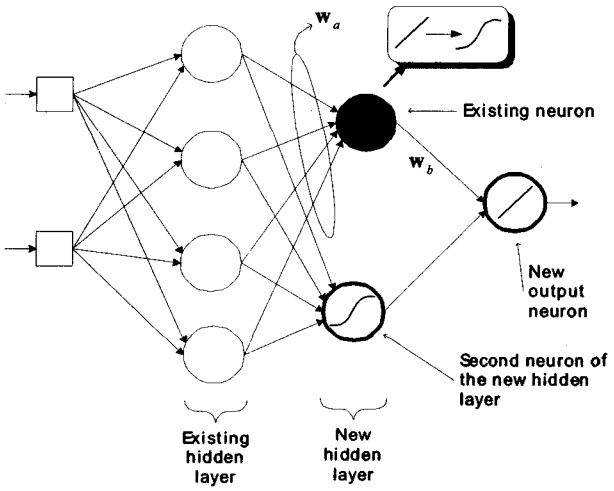


그림 5. 새로운 은닉층의 추가
Fig. 5. Addition of a new hidden layer.

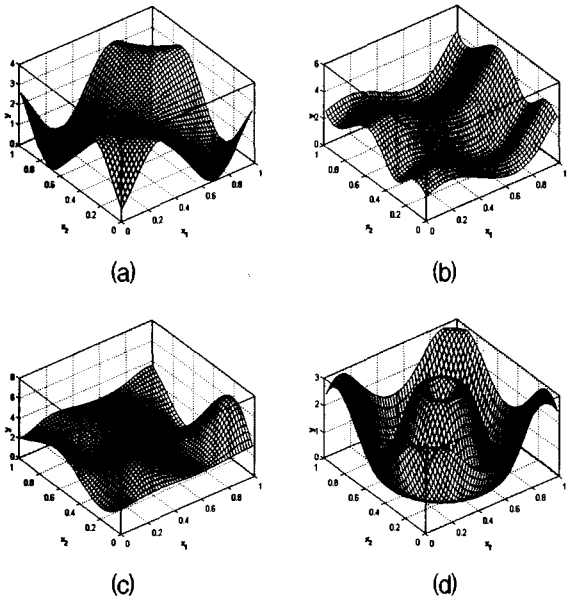


그림 6. 모의실험에 사용한 함수들 (a) g_1 (b) g_2 (c) g_3 (d) g_4
Fig. 6. Test functions for simulation (a) g_1, (b) g_2, (c) g_3 and (d) g_4.

225개의 학습 데이터는 다음과 같은 방법으로 생성되었다.

1) 정의역 $[0,1]^2$을 15×15의 균일한 사각형으로 나눈다.
2) 각각의 사각형에서 한 개의 점을 임의로 선택한다.

일반화 성능 테스트를 위한 데이터는 10,000개의 점을 균일하게 선택하였다.

신경회로망의 성능 지표로는 다음과 같이 정의되는 NFVU (normalized fraction of variance unexplain -ed)[13]를 사용하였다.

$$\mathrm{NFVU} = \sqrt{\frac{\sum_m (\hat{g}(\mathbf{x}_m) - g(\mathbf{x}_m))^2}{\sum_m (g(\mathbf{x}_m) - \bar{g})^2}} \quad (8)$$

여기서 $g(\mathbf{x}_m)$은 m번째 학습데이터 $\mathbf{x}_m$에 대한 원하는 출력값(desired output), $\hat{g}(\mathbf{x}_m)$는 $\mathbf{x}_m$에 대한 실제 신경회로망의 출력값, $\bar{g}$는 목표데이터(target data)의 평균이다. NFVU의 분모는 목표데이터의 분산이고, 분자는 일반적으로 쓰이는 MSE (mean-squared -error)이다. 실험에서 목표 NFVU는 0.04로 하였다.

실험결과의 성능 지표로는 테스트 데이터에 대해 NFVU로 나타낸 일반화 성능과 자유도(DoFs: degrees of freedom)를 사용하였다. DoFs는 신경회로망의 복잡도를 나타내는 것으로서 가중치 파라미터의 개수이다.

II장에서 서술한 알고리즘인 SOA-P와 SOA-D를 Setiono[5]의 DNC 알고리즘과 비교하였다. 이 알고리즘에서는 학습방법으로 근사뉴튼(quasi-Newton) 알고리즘을 이용하는데, 제안된 알고리즘과의 공정한 비교를 위해 학습알고리즘을 LM 알고리즘으로 한 경우도 함께 실험하였다. 또한 주어진 문제에 대한 최소구조를 찾는 성능을 알아보기 위해 고정된 크기를 가지는 신경회로망과 비교하였다. 이 경우 은닉뉴런의 수를 변화시켜가면서 각 함수별로 랜덤 초기 가중치가 다른 100개의 신경회로망을 생성하여 목표 NFVU인 0.04에 도달할 때까지 LM 알고리즘으로 학습시켰다. 초기 가중치는 [-0.05, +0.05]의 범위에서 무작위로 결정하였다. 최대 학습 수는 1200으로 하였다.

SOA-P와 SOA-D의 초기 네트워크는 18개의 후보 중에서 선택하였으며 (즉, N_c=18), SOA-P의 경우 은닉뉴런의 추가를 위해 5개의 후보를 사용하였다. 이 값들은 여러 경우 중에서 실험적으로 좋은 결과를 얻은 경우를 선택한 것이며 더 많은 수의 후보를 사용하는 것은 성능향상에 큰 도움을 주지 못하였다. 구조 증가 단계에서는 최대 100번의 학습을 하였고 뉴런 소거 단계에서는 최대 1200번의 학습을 하였다*. 초기 가중치의

* 10번의 실험에서 모두 같은 크기의 신경회로망을 생성하였기 때문에 평균과 가장 좋은 결과의 DoFs가 같다.

표 1. 제안된 알고리즘과 DNC의 비교. 10번의 실험에 대한 평균과 표준편차를 나타내었다.
Table 1. Comparisons of the proposed algorithms and DNC. The average and the standard deviation values for 10 trials are shown.

	Measures	DNC-QN	DNC-LM	Two-layer networks		Three-layer networks	
				SOA-P	SOA-D	SOA-P	SOA-D
g_1	DoFs	56.6	53.0	39.4	37	26.2	25
	(std)	(9.32)	(13.2)	(2.64)		(4.79)	
	Test NFVU	0.0566	0.0516	0.0497	0.0479	0.0411	0.0401
	(std)	(0.01082)	(0.00390)	(0.00144)		(0.00187)	
g_2	DoFs	32.2	27.8	23.4	29	-	-
	(std)	(5.6)	(3.80)	(3.68)			
	Test NFVU	0.0414	0.0423	0.0420	0.0384	-	-
	(std)	(0.00295)	(0.00298)	(0.00178)			
g_3	DoFs	77.4	57.0	49.0	49	48.0	51
	(std)	(13.2)	(6.52)	(0)		(3.69)	
	Test NFVU	0.0678	0.0633	0.0548	0.0541	0.0594	0.0564
	(std)	(0.00878)	(0.00674)	(0.00064)		(0.00491)	
g_4	DoFs	162.2	113.8	90.2	89	34.1	25
	(std)	(24.8)	(10.3)	(3.60)		(11.85)	
	Test NFVU	0.0690	0.0697	0.0604	0.0535	0.0439	0.0309
	(std)	(0.01605)	(0.02373)	(0.00758)		(0.00873)	

표 2. 제안된 알고리즘에 의해 생성된 2층 신경회로망과 고정된 구조를 가지는 신경회로망의 비교. 고정구조 신경회로망의 경우, 100번의 시도에 대한 학습 성공률을 ( ) 안에 나타내었다.
Table 2. Comparisons of the performance of the two-layer networks by the proposed algorithms with that of the networks having fixed structures. For the fixed topology the success rate of 100 trials is given in ( ).

	Measures	Fixed topology				SOA-P avg.	SOA-P best	SOA-D
g_1	DoFs	29 (0%)	33 (2%)	37 (11%)	41 (52%)	39.4	37	37
	Structure	2-7-1	2-8-1	2-9-1	2-10-1	-	2-9-1	2-9-1
	Test NFVU	-	0.0513	0.0497	0.0510	0.0497	0.0482	0.0479
g_2	DoFs	17 (0%)	21 (4%)	25 (92%)	29 (96%)	23.4	21	29
	Structure	2-4-1	2-5-1	2-6-1	2-7-1	-	2-5-1	2-7-1
	Test NFVU	-	0.0430	0.0411	0.0411	0.0420	0.0384	0.0384
g_3	DoFs	41 (0%)	45 (3%)	49 (21%)	53 (48%)	49.0	49**	49
	Structure	2-10-1	2-11-1	2-12-1	2-13-1	-	2-12-1	2-12-1
	Test NFVU	-	0.0627	0.0598	0.0617	0.0548	0.0542	0.0541
g_4	DoFs	85 (0%)	89 (8%)	93 (38%)	97 (65%)	90.2	85	89
	Structure	2-21-1	2-22-1	2-23-1	2-24-1	-	2-21-1	2-22-1
	Test NFVU	-	0.0639	0.0615	0.0611	0.0604	0.0519	0.0535

크기는 [-0.05, +0.05]의 범위에서 결정하였다. 모든 실험은 10번을 반복 수행하였다.

표 1은 근사뉴튼 학습방법을 이용한 DNC (DNC-QN) 및 LM 알고리즘에 의한 학습을 이용한 DNC (DNC-LM)와 제안된 알고리즘을 비교한 것이다. 은닉 뉴런의 개수와 일반화 성능, 그리고 10번의 실험에 대한 각각의 표준편차를 함께 나타내었다. (SOA-D의 경우는 난수발생과정이 없어 항상 같은 결과를 보이므로 표준편차는 0이며 생략하였다.) 제안된 알고리즘의 경우 층의 증가를 허용하지 않은 경우와 허용하여 3층 신경회로망을 구성하도록 한 경우를 함께 나타내었다. g_2의 경우는 함수가 간단하여 층의 증가가 일어나지 않았기 때문에 3층 신경회로망의 경우에서 제외되었다. 제안된 알고리즘에 의한 2층 신경회로망은 전반적으로 DNC에 비해 우수한 성능을 보인다. 신경회로망의 크기 측면에서 DNC-LM이 g_2에 대해 SOA-D보다 약간 작

** 더 많은 학습을 반복해도 오차가 감소하지 않는 것을 관찰하였기 때문에 이와 같이 설정하였다.

표 3. 제안된 알고리즘에 의해 생성된 3층 신경회로망과 고정된 구조를 가지는 신경회로망의 비교. 고정구조 신경회로망의 경우, 100번의 시도에 대한 학습 성공률을 ( ) 안에 나타내었다.

Table 3. Comparisons of the performance of the three-layer networks by the proposed algorithms with that of the networks having fixed structures. For the fixed topology the success rate of 100 trials is given in ( ).

	Measures	Fixed topology				SOA-P avg.	SOA-P best	SOA-D
g_1	DoFs	19 (0%)	20 (2%)	25 (2%)	30 (71%)	26.2	20	25
	Structure	2-2-3-1	2-3-2-1	2-4-2-1	2-5-2-1	-	2-3-2-1	2-4-2-1
	Test NFVU	-	0.0376	0.0400	0.0460	0.0411	0.0384	0.0401
g_3	DoFs	37 (0%)	43 (8%)	44 (25%)	51 (73%)	48.0	43	51
	Structure	2-4-4-1	2-6-3-1	2-5-4-1	2-5-5-1	-	2-6-3-1	2-5-5-1
	Test NFVU	-	0.0554	0.0575	0.0582	0.0594	0.0499	0.0564
g_4	DoFs	20 (0%)	25 (3%)	31 (6%)	44 (15%)	34.1	25	25
	Structure	2-3-2-1	2-3-3-1	2-4-3-1	2-5-4-1	-	2-3-3-1	2-3-3-1
	Test NFVU	-	0.0415	0.5029	0.2921	0.0439	0.0328	0.0309

은 것을 제외하고는 모든 경우 제안된 알고리즘이 DNC-QN에 비해서는 약 10~45%, DNC-LM에 비해서는 15~26% 정도 더 작은 구조를 찾았다.

일반화 성능의 경우, g_2에 대해 DNC 방법이 SOA-P와 비슷한 것을 제외하고는 제안된 방법이 약 10~15% 정도 낮은 NFVU를 보인다. 또한 제안된 알고리즘이 DoFs와 일반화성능 모두에서 훨씬 더 작은 표준편차를 보여 알고리즘의 신뢰성 측면에서 향상된 것을 볼 수 있다. 3층 신경회로망의 결과 역시 DNC의 결과와 비교하여 구조, 일반화성능, 신뢰성 측면에서 우수하다. 특히 g_1과 g_4의 경우 2층 신경회로망에 대한 제안된 알고리즘보다도 더 작고 일반화성능이 우수한 신경회로망을 구성할 수 있었다. 이는 제안된 알고리즘이 층의 증가 과정을 통해 더 작은 구조의 신경회로망을 구성할 수 있는 가능성이 있음을 보여준다.

표 2는 층의 증가를 허용하지 않았을 경우 제안된 알고리즘과 고정구조 신경회로망을 비교한 것이다. SOA-P에 의한 최소 신경회로망의 크기는 (표에서 SOA-P best 항목) 학습 가능한 고정구조 신경회로망의 최소구조에서 크게 벗어나지 않는다. g_4의 경우 고정된 구조에서는 학습이 불가능한 2-21-1 구조의 신경회로망을 생성했으며, g_2의 경우 가장 작은 학습 가능한 구조를, g_1와 g_3의 경우에는 최소구조보다 하나 많은 은닉뉴런을 갖는 신경회로망을 얻을 수 있었다. SOA-P의 평균적인 신경회로망 구조는 SOA-P의 최소구조보다 약간 많은 경향이지만 역시 최소구조에 가까운 것을 알 수 있다. 그리고 일반화 성능 역시 고정구조 신경회로망보다 비슷하거나 좋은 것을 볼 수 있다. SOA-D는 SOA-P와 비교했을 때 g_2와 g_4에 대해 한두 개 많은 은닉뉴런을 신경회로망을 생성하였으나 역시 최소구조에 가까우며, 일반화 성능 또한 고정구조에 비해 우수하다.

그림 7은 2층 신경회로망에 대한 자기구성과정의 학습곡선을 나타낸 것이다. 학습의 후반부에 나타나는 피크는 은닉뉴런을 제거할 때 발생한 것이다. 각 학습곡선의 앞부분에서는 계단현상을 볼 수 있는데, 이것은 학습이 되지 않는 상태의 신경회로망에 뉴런을 첨가함으로서 오차가 급격히 감소하는 현상이다. 두 알고리즘에 의한 전체적인 학습곡선이 유사한 경향을 보이고 있다.

표 3은 층의 증가를 허용하여 2개의 은닉층을 갖도록 한 경우의 결과를 비교한 것이다. 층의 증가를 허용하였을 경우의 결과는 2층 신경회로망을 구성하였을 때와 같은 경향을 보인다. SOA-P에 의한 신경회로망의 최소구조는 학습 가능한 최소 구조의 고정구조 신경회로망과 같음을 알 수 있으며, 고정구조에서 이들 구조의 학습 성공률은 각각 2%, 8%, 3%에 불과하다. SOA-D의 결과도 SOA-P의 가장 좋은 결과와 비슷하다. g_4의 경우 SOA-P의 최소구조와 같은 결과를, g_1, g_3의 경우 SOA-P에 의한 최소구조보다 한 개 많은 뉴런을 가지는 신경회로망이 생성되었다. 두 알고리즘의 테스트 NFVU는 고정구조의 경우와 비슷하거나 더 낮은 것을 알 수 있다. SOA-P와 SOA-D의 성능의 우수함은 g_4의 경우 특히 두드러지게 나타났다. 두 알고리즘에 의한 신경회로망의 최소구조는 고정구조에서 학습 성공률이 3%에 불과한 것이며, 일반화성능을 비교했을 때 고

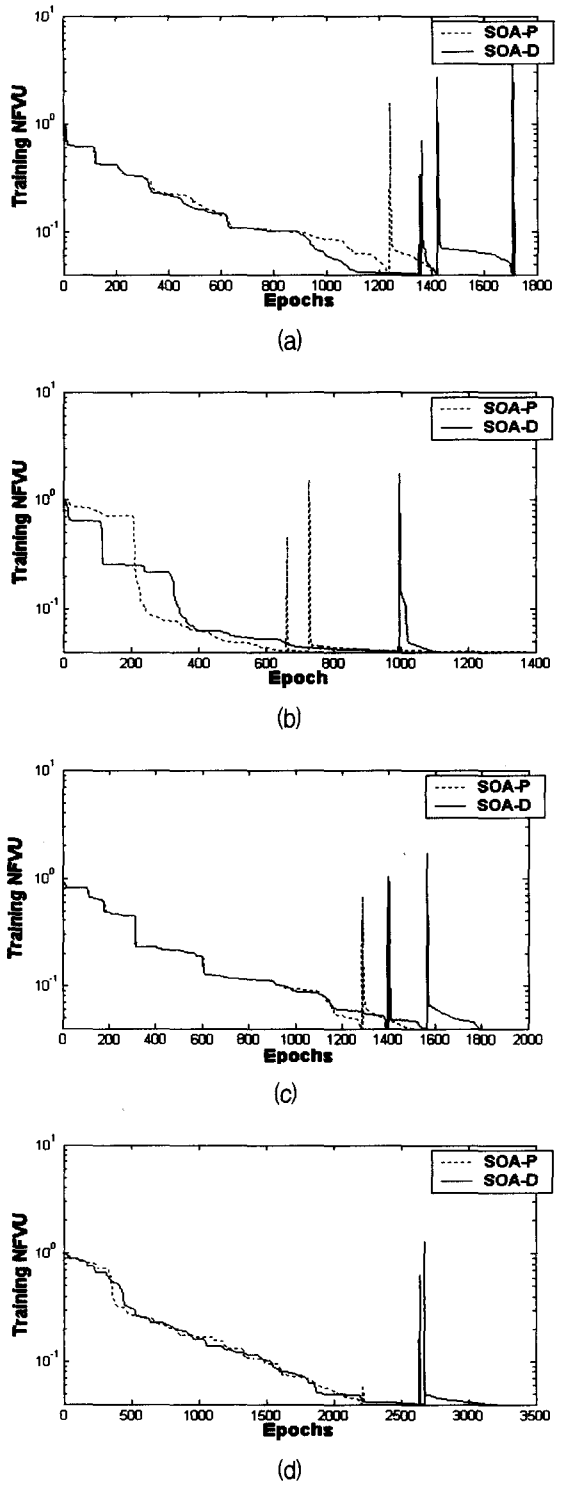


그림 7. 2층 신경회로망 학습곡선 (a) g_1 (b) g_2 (c) g_3 (d) g_4

Fig. 7. Learning curves of 2-layer networks for (a) g_1, (b) g_2, (c) g_3 and (d) g_4.

정구조에서 발생하는 과도학습현상 없이 낮은 오차값을 내는 것을 볼 수 있다.

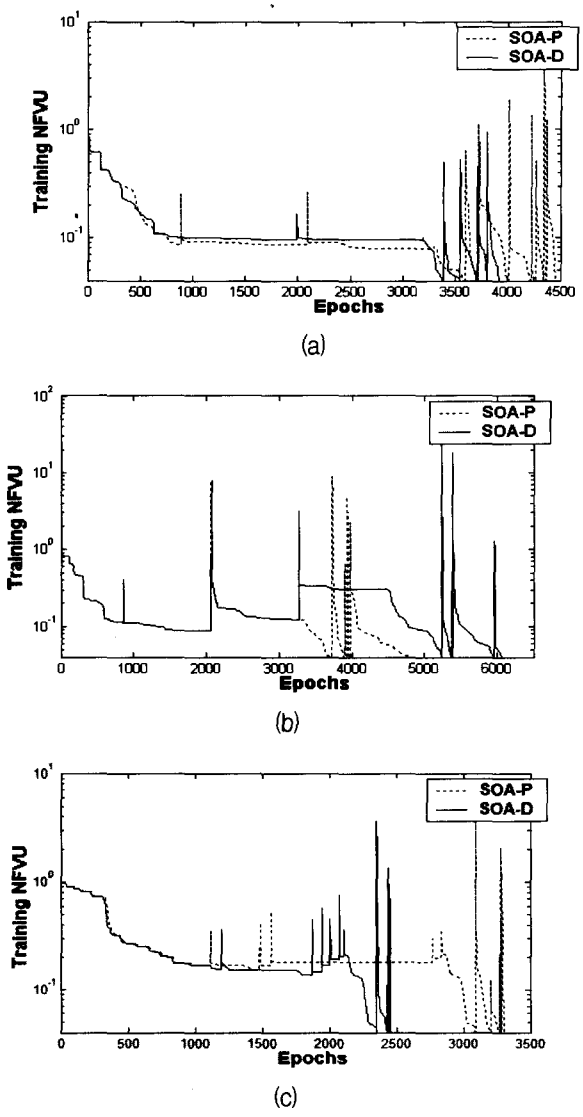


그림 8. 3층 신경회로망 학습곡선 (a) g_1 (b) g_3 (c) g_4

Fig. 8. Learning curves of 3-layer networks for (a) g_1, (b) g_3 and (c) g_4.

그림 8은 3층 신경회로망에 대한 자기구성과정의 학습곡선을 나타낸 것이다. 학습의 중간에서 평평한 부분과 피크가 번갈아 나타나는 부분은 층의 추가 전에 뉴런을 제거하는 과정이다. 층의 증가를 허용하지 않은 그림 7과 비교했을 때 전체적인 학습 수는 2~3배 정도 증가하였다. 이는 층의 증가 전의 뉴런제거 과정에서 많은 시간이 소요되었기 때문이다.

SOA-P 및 SOA-D에서 가장 시간이 많이 소요되는 것은 연결소거 과정이다. 이 때문에 연결소거 과정이 없는 기존의 DNC 방법에 비해 그만큼 더 많은 시간과 계산량을 필요로 한다. 초기 네트워크를 선택하는데 걸리는 시간이나 SOA-P에서의 후보집합 구성 및 SOA

-D에서의 새로운 가중치값 계산에 걸리는 시간은 전체 신경회로망 구성과정의 10% 미만으로 측정되었다. LM 알고리즘으로 학습하는 DNC에 비해 제안된 알고리즘은 층의 증가를 허용하지 않는 경우 평균 1.5~2배 정도, 층의 증가를 허용하는 경우 2~4배 정도의 시간을 요하는 것을 관측하였다*. 그러나 표 1에서 나타난 성능의 향상을 고려할 때, 이러한 계산량 증가는 가치가 있다고 할 수 있다.

Ⅳ. 결 론

본 논문에서는 다층 신경회로망을 이용한 자기구성 알고리즘을 제안하였으며 모의실험을 통하여 그 성능을 기존의 DNC 알고리즘 및 고정구조 신경회로망과 비교 측정하였다. 개발된 알고리즘의 중요한 특성은 좋은 일반화 성능을 갖는 학습가능한 작은 크기의 최적화된 신경회로망을 신뢰성 있게 구하는 것이다. 이를 위해 구조증가 알고리즘과 연결소거 알고리즘을 같이 사용하였다. DNC 알고리즘에 비해 신경회로망의 크기, 일반화 성능 및 신뢰성 측면에서 향상된 결과를 나타내었으며, 고정된 구조를 가지는 신경회로망을 학습시킨 것과 비교하였을 때 좋은 일반화 성능을 가지면서 학습가능한 최소구조에 가까운 신경회로망을 구성할 수 있는 것으로 나타났다.

추후과제로는 제안한 알고리즘을 함수 근사화 문제에서 입력 차원이 높은 경우와 패턴 인식 문제에 적용하는 것을 들 수 있다. 또한, 제안한 알고리즘에 기초하여 다른 활성함수를 가지는 신경회로망에 대한 알고리즘으로 확장하고자 한다.

참 고 문 헌

[1] S. Haykin, *Neural Networks: A Comprehensive Foundation*, NJ: Prentice-Hall, 1999.

[2] M. T. Hagan, H. B. Demuth, and M. Beale, *Neural Network Design*, Boston, MA: PWS Publishing, 1996.

[3] T.-Y. Kwok and D.-Y. Yeung, "Constructive algorithms for structure learning in feedforward neural networks for regression problems," *IEEE Trans. Neural Networks*, vol. 8, no. 3, pp. 630-645, May, 1997.

[4] T. Ash, "Dynamic node creation in backpropagation networks," *Connection Sci.*, vol. 1, no. 4, pp. 365-375, 1989.

[5] R. Setiono and L. C. K. Hui, "Use of a quasi-Newton method in a feedforward neural network construction algorithm," *IEEE Trans. Neural Networks*, vol. 6, no. 1, pp. 273-277, Jan. 1995.

[6] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Trans. Neural Networks*, vol. 5, no. 6, pp. 989-993, Nov. 1994.

[7] J.-S. Lee and C. H. Park, "Self-organizing neural network using adaptive neurons," in *Proc. Int. Conf. Neural Information Processing*, Singapore, pp. 935-939, Nov. 2002.

[8] K. Hornik, M. Stinchdombe, and H. White, "Multilayer feedforward networks are universal approximaters," *Neural Networks*, vol. 2, pp. 359-366, 1989.

[9] S. Tamura and M. Tateishi, "Capabilities of a four-layered feedforward neural network: four layers versus three," *IEEE Trans. Neural Networks*, vol. 8, pp. 251-255, Mar. 1997.

[10] L. Breiman "The pi method for estimating multivariate functions from noisy data," *Technometrics*, vol. 3, no. 2, pp. 125-160, 1991.

[11] M. Maechler, D. Martin, J. Schimert, M. Csoppensky and J. N. Hwang, "Project pursuit learning networks for regression," in *Proc. Int. Conf. Tools for AI*, Washington D.C., pp. 350-358, Nov. 1990.

[12] V. Cherkassky and H. Lari-Najafi, "Constrained topological mapping for nonparametric regression analysis," *Neural Networks*, vol. 4, pp. 27-40, 1991.

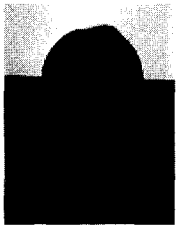
[13] J. H. Friedman, "Classification and multiple regression through projection pursuit," Dept.

* DNC-QN과 제안된 알고리즘은 학습방법이 다르기 때문에 비교하기에 적합하지 않다. 실제로 근사뉴튼 알고리즘이 LM 알고리즘보다 수렴속도가 느려 DNC-QN은 제안된 방법보다 더 긴 시간이 걸린다.

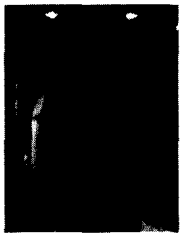
Statistics, Stanford Univ. Technical Report, no. 12, Jan. 1985.

저 자 소 개

이 종 석(학생회원)
1999년 한국과학기술원 전기 및 전자공학과 학사 졸업.
2001년 한국과학기술원 전자전산학과 석사 졸업.
2004년 현재 한국과학기술원 전자전산학과 박사과정.
<주관심분야: 신경회로망, 바이모달 음성인식>



정 승 범(정회원)
1995년 한국과학기술원 전기 및 전자공학과 학사 졸업.
1998년 한국과학기술원 전기 및 전자공학과 석사 졸업.
현재 삼성전자 기술총괄 소프트웨어 센터 연구원.
<주관심분야: 디지털 신호처리>



김 재 영(정회원)
1996년 한국과학기술원 전기 및 전자공학과 학사 졸업.
1998년 한국과학기술원 전기 및 전자공학과 석사 졸업.
1998년~2001년 서두인칩 근무.
2001년~2003년 GCT Semiconductor 근무
현재 서두인칩 연구원.
<주관심분야: IEEE802.16d,e, High-speed portable internet>

박 철 훈(정회원)
1984년 서울대학교 전자공학과 학사 졸업.
1985년 캘리포니아 공과대학교 전기공학과 석사 졸업.
1990년 캘리포니아 공과대학교 전기공학과 박사 졸업.
1991년~1992년 한국과학기술원 전기 및 전자공학과 대우교수/연수 연구원.
현재 한국과학기술원 전자전산학과 교수.
미국 전기전자공학회 준석학회원.
<주관심분야: 지능시스템, 정보처리, 지능제어, 최적화>