한국지능시스템학회 논문지 2009, Vol. 19, No. 4, pp. 574-579

Flexible Incremental 알고리즘을 이용한 신경망의 단계적 구축 방법

Stepwise Constructive Method for Neural Networks Using a Flexible Incremental Algorithm

박진일* · 정지석* · 조영임** · 전명근*

Jin-Il Park*, Ji-Suk Jung*, Young-Im Cho** and Myung-Geun Chun*

* 충북대학교 전자정보대학 제어로봇공학과

** 수원대학교 IT대학 컴퓨터학과

요 약

복잡한 비선형 회귀문제들에서 최적의 신경망을 구축하기 위해서는 구조의 선정 및 노이즈에 의한 과잉학습(overtraining) 등에 따른 많은 문제들이 있다. 본 논문에서는 flexible incremental 알고리즘을 이용하여 단계적으로 최적의 신경망을 구축하는 방법을 제안한다. Flexible incremental 알고리즘은 예측 잔류오차를 최소화하기 위해 단계적으로 추가되어지는 은닉노드 개수를 검증데이터를 이용하여 신축성 있게 조절하고, 빠른 학습을 위하여 ELM (Extreme Learning Machine)을 이용한다. 제안된 방법은 신경망의 구축과정에서 사용자의 어떠한 관여 없이도 빠른 학습과 적은 수의 은닉노드들에 의한 범용 근사화 (universal approximation)가 가능한 신경망의 구축이 가능한 장점을 가지고 있다. 다양한 종류의 벤치마크 데이터들을 이용한 실험 결과를 통하여 제안된 방법이 실제 회귀문제들에서 우수한 성능을 가짐을 확인하였다.

키워드 : Flexible incremental algorithm, 단계적 구축 방법, ELM(Extreme Learning Machine), 과잉학습.

Abstract

There have been much difficulties to construct an optimized neural network in complex nonlinear regression problems such as selecting the networks structure and avoiding overtraining problem generated by noise. In this paper, we propose a stepwise constructive method for neural networks using a flexible incremental algorithm. When the hidden nodes are added, the flexible incremental algorithm adaptively controls the number of hidden nodes by a validation dataset for minimizing the prediction residual error. Here, the ELM (Extreme Learning Machine) was used for fast training. The proposed neural network can be an universal approximator without user intervene in the training process, but also it has faster training and smaller number of hidden nodes. From the experimental results with various benchmark datasets, the proposed method shows better performance for real-world regression problems than previous methods.

Key Words : Flexible incremental algorithm, Stepwise constructive method, ELM, Overtraining.

1. 서 론

신경망은 복잡한 비선형 함수들을 근사화 하거나 모델들을 예측하는 문제들에서 우수한 성능을 가지고 있어 지금까지 많은 분야에 적용되고 있다. 그러나 취득된 데이터에 최적의 신경망을 구축하기 위해서는 데이터의 특성에 적합한 신경망 구조의 선정과 노이즈에 의한 과잉학습(overtraining)등과 같이 아직도 명확히 해결되지 않은 문제들 남아있다. 이와 관련된 대표적인 연구들로는 cross validation [1], node pruning [2] 등이 있다. cross validation은 취득된 데이터를 학습과 검증 및 테스트 데이터로 구분하는 방법으로 학습과 검증데이터를 적절하게 분류하는 문제와 분류된 데이터들이 우수한 수렴 성능을 가질 수 있는가에 대해서는 명확히 설명하기가 어려운 문제점이 있다. node pruning은 학습을 통하여 구축되어진 신경망에서 적절하지 않은 노드들을 제거하는 방법으로 이러한 방법은 사전에 미리 구축되어 있는 신경망을 고려한다는 점에서 앞선 방법과 차이가 있다.

Huang등은 단일 은닉층을 가지는 전방향 신경망의 학습방법으로 단 한번의 전방향 단계에서 학습이 종료됨으로써 오류 역전파 (BP: Back-Propagation) 알고리즘에 비하여 학습시간이 월등히 우수한 ELM (Extreme Learning Machine)을 제안하였다 [3]. ELM 알고리즘은 전방향 신경망의 빠른 학습시간을 위하여 입력층과 은닉층을 연결하는 입력 가중치들과 은닉노드들의 바이어스 값들을 랜덤하게 생성한 후 놈 최소자승해 (norm least-squares solution)와 모어-페로스의 일반화된 역행렬 (Moore-Penrose general-

접수일자 : 2009년 1월 19일
완료일자 : 2009년 7월 31일
이 논문은 2009년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2009-0076094).

ized inverse)을 이용하여 은닉층과 출력층을 연결하는 출력 가중치들을 계산함으로써 빠른 시간에 우수한 성능을 얻을 수 있는 장점을 가지고 있다. Han과 Huang은 ELM의 함수 근사화에 사전 정보를 고려하는 방안으로 Fourier series를 기반으로 은닉노드들의 활성화함수로 sine과 cosine 함수들을 이용하는 SCELM (Sine and Cosine ELM)을 제안하였다 [4]. Liang과 Huang은 ELM을 연속적인 데이터 입력에 따른 재귀적 최적화 방법으로 OS-ELM (Online Sequential ELM)을 제안하였다 [5]. Huynh과 Won은 기존 ELM이 입력 가중치들과 바이어스들을 랜덤하게 선택하는 것에 반하여 선형모델을 기반으로 출력 가중치들을 랜덤하게 선정한 후에 역으로 입력 가중치들과 바이어스들을 계산하고 다시 출력 가중치들을 갱신하는 LS-ELM (Least-Squares ELM)을 제안하였다 [6].

위와 같이 ELM에 기반을 둔 방법들은 신경망의 구축시간이 빠르고, 일반적인 성능이 우수한 장점을 가지고 있지만, 사전에 사용자에 의하여 신경망 구축에 필요한 은닉노드 개수를 선정하는 문제와 복잡한 비선형 회귀문제들에서 과잉학습의 영향을 해결해야 하는 문제점들을 여전히 가지고 있다 [7]. Huang 등은 단일 은닉층을 가지는 전방향 신경망의 은닉노드 개수를 단계적으로 1개씩 추가하는 I-ELM (Incremental ELM)을 제안하였다 [8]. I-ELM은 신경망을 구축하기위해 사전에 은닉노드 개수를 결정해야 하는 어려움을 극복하고, 구축된 신경망이 범용 근사화 (universal approximation)가 가능함을 이론과 실험을 통하여 증명되었다 [8]. 그러나 I-ELM은 점증적으로 하나의 은닉노드를 추가함에 따라 원하는 수준의 성능을 얻기까지 많은 수의 은닉노드들과 오랜 구축 시간이 필요하다는 단점을 가지고 있다. 본 논문에서는 추가되어지는 은닉노드 개수를 flexible incremental 알고리즘을 이용하여 신축성 있게 조절하는 방법 (FI-ELM: Flexible I-ELM)을 제안한다.

본 논문의 구성은 2장에서 I-ELM을 이용한 단일 은닉층을 가지는 전방향 신경망의 증분 알고리즘에 관하여 소개하고, 3장에서 제안된 방법에 관하여 설명한다. 4장에서는 실제 벤치마크 데이터들을 이용하여 실험한 결과를 논하고, 마지막으로 5장에서 마무리를 한다.

2. ELM을 이용한 단일 은닉층을 가지는 전방향 신경망의 증분 알고리즘

그림 1은 단일 은닉층을 가지는 I-ELM의 구조를 보여준다.

n개의 은닉노드들에 의한 단일 은닉층을 가지는 신경망의 출력 함수는 다음과 같이 표현되어질 수 있다.

$$f_n(\mathbf{x}) = \sum_{i=1}^{n} \beta_i g_i(\mathbf{x}) = \sum_{i=1}^{n} \beta_i G(\mathbf{x}, \mathbf{a}_i, b_i) \quad (1)$$
$$\mathbf{a}_i \in \mathbf{C}^{\mathbf{d}}, \mathbf{x} \in \mathbf{C}^{\mathbf{d}}, b_i \in \mathbf{C}, \beta_i \in \mathbf{C}$$

여기서 g_i또는 $G(\mathbf{x}, \mathbf{a}_i, b_i)$는 i번째 은닉노드의 출력함수이고, β_i는 i번째 은닉노드와 출력 노드사이의 출력 가중치이다.

신경망의 출력 함수 f_n과 목적 함수 f의 유사도는 다음과 같이 구해 질 수 있다.

$$\| f_n - f \| = \left[\int_X (f_n(\mathbf{x}) - f(\mathbf{x})) \overline{(f_n(\mathbf{x}) - f(\mathbf{x}))} d\mathbf{x} \right]^{1/2} \quad (2)$$

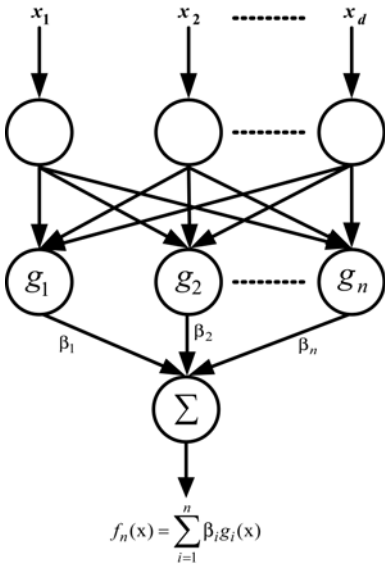

그림 1. I-ELM의 구조
Fig. 1. The structure of I-ELM

n개의 은닉노드들에 의한 신경망의 잔류오차 (residual error)는 $e_n = f - f_n$이 된다. 단계적으로 랜덤하게 추가되어지는 n번째 은닉노드에서의 출력 가중치 β_n는 다음과 같이 계산되어진다.

$$\beta_n = \frac{< e_{n-1}, g_n >}{\| g_n \|^2} \quad (3)$$

따라서 잔류 오차 e_n은 $e_n = f - f_n = e_{n-1} - \beta_n g_n$이 되며, e_n과 g_n의 내적은 다음과 같다.

$$\begin{aligned} < e_n, g_n > &= < e_{n-1} - \beta_n g_n, g_n > \\ &= < e_{n-1}, g_n > - \beta_n \| g_n \|^2 \end{aligned} \quad (4)$$

위의 식 (3)과 (4)로부터

$$\begin{aligned} < e_n, g_n > &= < e_{n-1}, g_n > - \frac{< e_{n-1}, g_n >}{\| g_n \|^2} \cdot \| g_n \|^2 \\ &= < e_{n-1}, g_n > - < e_{n-1}, g_n > = 0 \end{aligned} \quad (5)$$

따라서 $< e_n, e_n - e_{n-1} > = < e_n, -\beta_n g_n > = 0$ 이 되며 이것은 $e_n \perp (e_n - e_{n-1})$를 의미한다.

I-ELM은 단순히 은닉노드들을 랜덤하게 선택하고, 추가되는 은닉노드들의 출력 가중치를 식 (3)과 같이 계산하는 것에 의하여 예측 잔류오차 e_n을 0으로 수렴하게 된다. Huang등은 이러한 증분 알고리즘에 의하여 구축된 I-ELM이 범용 근사화가 가능함을 이론과 실험을 통하여 증명하였다 [8]. 그러나 I-ELM은 원하는 성능을 얻기까지 많은 개수의 은닉노드들과 오랜 구축 시간이 필요하다는 단점을 가지고 있다. 특히 적절하지 못한 은닉노드의 추가는 신경망의 출력에서 매우 적은 영향을 미치게 되고 결과적으로 신경망의 복잡성을 증가시킬 수 있다.

3. Flexible incremental 알고리즘을 이용한 신경망의 단계적 구축방법

신경망에서 시스템에 적합한 은닉층의 개수와 은닉노드 개수를 결정하는 것은 신경망의 성능에 있어서 중요한 이슈로 자리 잡고 있다. 신경망에서 너무 적은 개수의 은닉노드들은 예측하고자 모델을 정확하게 표현하기가 어려운 반면, 너무 많은 은닉노드 개수의 확장은 테스트 데이터에 대한 과잉학습의 우려와 연산시간의 효율 면에서 좋지 않은 결과를 얻을 수 있다. 그림 2는 ELM에서 은닉노드 개수에 따른 과잉학습의 한 예를 보여준다.

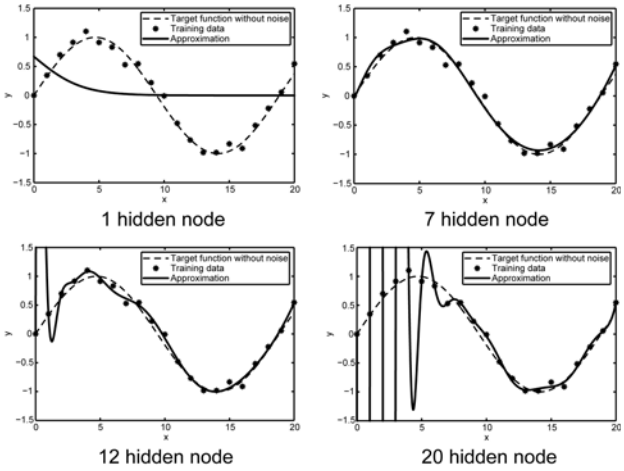

그림 2. 은닉노드 개수의 증가에 따른 과잉학습의 예
Fig. 2. An example of the overtraining by the number of hidden nodes

그림 2에서 훈련데이터는 21개로 $y=\sin(x/3)+v$에서 추출하였다. 여기서 v는 균일 분포 랜덤 변수(uniformly distributed random variable)로 -0.2에서 0.2사이의 값을 가진다. 테스트 데이터는 v를 제외하고 실험하였다. 그림 2에서 은닉노드 개수가 1개인 경우 구축된 신경망은 훈련 및 테스트 데이터에서를 근사화하기에 부족하며, 은닉노드 개수가 7개인 경우 우수한 근사화 성능을 얻을 수 있음을 알 수 있다. 그러나 은닉노드 개수가 12개인 경우 테스트 데이터를 통하여 과잉학습으로 인한 영향을 볼 수 있으며, 은닉노드 개수가 20개인 경우 과잉학습으로 인한 영향을 확연히 확인할 수 있다. 따라서 너무 적은 개수의 은닉노드들은 근사화 성능이 부족하지만 적절한 개수의 은닉노드들로 구축된 신경망은 우수한 성능을 얻을 수 있다. 하지만, 너무 과도한 개수의 은닉노드들은 신경망의 성능에 좋지 못한 영향을 줄 수 있다.

본 논문에서는 검증데이터를 이용하여 추가되는 은닉노들의 과잉학습에 따른 영향을 예측하여 적절한 개수의 은닉노드들을 추가하고, 빠른 학습을 위하여 ELM을 이용하는 방법 (FI-ELM: Flexible I-ELM)을 제안한다.

제안하는 FI-ELM을 각 단계별로 설명하면 다음과 같다.

훈련데이터 $\aleph=\{(\mathbf{x}_i, t_i)|\mathbf{x}_i\in\mathbf{C}^d, t_i\in\mathbf{C}, i=1,...,N\}$일때, 최대 은닉노드 개수 $L_{\max}$와 허용 오차 ϵ를 설정한다.

[단계 1] 훈련데이터 $\aleph$를 50%는 학습데이터 $\mathbf{l}$로 나머지 50%는 검증데이터 $\mathbf{v}$로 분류한다.

[단계 2] 초기 은닉노드 개수 $L=0$, 학습데이터의 잔류 오차 $E^{\mathbf{l}}_{old}=t^{\mathbf{l}}$, 검증데이터의 잔류 오차 $E^{\mathbf{v}}_{old}=t^{\mathbf{v}}$로 설정한다.

[단계 3] 은닉노드 개수를 1개 증가 시킨다 (L:L=L+SL, SL=1). 여기서 SL은 새로 추가되어질 은닉노드 개수로 [단계 6]~[단계 7]에 의해 최종적으로 결정되어 진다. 새로 추가되는 은닉노드의 입력 가중치들과 바이어스를 랜덤하게 설정하고, 은닉층의 출력과 학습데이터의 $E^{\mathbf{l}}_{old}$을 이용하여 아래 식 (8)과 같이 출력 가중치 β_{SL}을 계산한다.

$$\beta_{SL}=\frac{E^{\mathbf{l}}_{old}\cdot H^T_{SL}}{H_{SL}\cdot H^T_{SL}} \tag{6}$$

[단계 4] 새로 추가된 은닉노드 SL에 따른 학습데이터의 잔류 오차 $E^{\mathbf{l}}$를 계산한다.

$$E^{\mathbf{l}}=E^{\mathbf{l}}_{old}-\beta_{SL}\cdot H_{SL} \tag{7}$$

학습데이터의 잔류 오차$E^{\mathbf{l}}_{old}$와 새로운 학습데이터의 잔류 오차 $E^{\mathbf{l}}_{new}$를 갱신한다.

$$E^{\mathbf{l}}_{new}=E^{\mathbf{l}}_{old}=E^{\mathbf{l}} \tag{8}$$

[단계 5] 검증데이터의 잔류 오차 $E^{\mathbf{v}}$를 계산한다.

$$E^{\mathbf{v}}=E^{\mathbf{v}}_{old}-\beta_{SL}\cdot H_{SL} \tag{9}$$

검증데이터의 잔류 오차 $E^{\mathbf{v}}_{old}$와 새로운 검증데이터의 잔류 오차 $E^{\mathbf{v}}_{new}$를 갱신한다.

$$E^{\mathbf{v}}_{new}=E^{\mathbf{v}}_{old}=E^{\mathbf{v}} \tag{10}$$

[단계 6] 새로 추가된 은닉노드 SL에 1개의 새로운 은닉노드를 랜덤하게 추가한다($SL=SL+1$). 다음 학습데이터 $\mathbf{l}$과 새로 추가된 은닉노드 SL을 이용하여 출력 가중치 β_{SL}을 다시 계산한다.

$$\beta_{SL}=\frac{E^{\mathbf{l}}_{old}\cdot H^T_{SL}}{H_{SL}\cdot H^T_{SL}} \tag{11}$$

[단계 7] 새로 추가된 은닉노드 SL에 따른 학습데이터의 잔류 오차 $E^{\mathbf{l}}_{new}$과 검증데이터의 잔류 오차 $E^{\mathbf{v}}_{new}$를 다시 계산한다.

$$\begin{aligned}E^{\mathbf{l}}_{new}&=E^{\mathbf{l}}_{old}-\beta_{SL}\cdot H_{SL}\\ E^{\mathbf{v}}_{new}&=E^{\mathbf{v}}_{old}-\beta_{SL}\cdot H_{SL}\end{aligned} \tag{12}$$

만일 $E^{\mathbf{l}}_{old}\geq E^{\mathbf{l}}_{new}$ 또는 $E^{\mathbf{v}}_{old}\geq E^{\mathbf{v}}_{new}$이면, 새로 추가된 은닉노드 SL은 과잉학습의 영향을 만족한다고 판단하고 $E^{\mathbf{l}}_{old}=E^{\mathbf{l}}_{new}$, $E^{\mathbf{v}}_{old}=E^{\mathbf{v}}_{new}$로 갱신한 후 [단계 6]으로 가서 새로운 1개의 은닉노드를 SL에 추가한다. 그렇지 않으면 새로 추가된 은닉노드 SL이 과잉학습으로 인하여 성능이 좋지 않을 것으로 판단하여 앞서 계산한 β_{SL}을 기존의 값으로 하고, $SL=SL-1$이 된다. 따라서 총 은닉노드 개수는 $L:L=L+SL$이 된다.

[단계 8] 만일 종료조건 ($L \geq L_{\max}$ 또는 $\|E\| \leq \varepsilon$)을 만족하면 정지하고, 그렇지 않은 경우 [단계 3]~[단계 8]을 반복 수행한다.

4. 실험 및 결과 고찰

본 논문에서는 제안된 방법(FI-ELM)의 타당성을 검증하기 위하여 표 1과 같이 일반적인 회귀문제에서 많이 사용되고 있는 UCI 벤치마크 데이터들을 이용하여 실험을 하였다[9]. FI-ELM에서 각각의 벤치마크 데이터들은 학습, 검증과 테스트 데이터로 구분하였으며, 이때 학습과 검증데이터는 기존의 훈련데이터를 50%로 나누어 학습과 검증데이터로 구분하였다. 테스트 데이터는 기존의 방법과 제안된 방법 모두 동일한 데이터를 이용하였다. 각각의 실험에서 신경망의 모든 입력데이터는 -1과 1사이의 값으로 정규화하였으며, 출력 데이터는 0과 1사이의 값으로 정규화 하였다. 실험에 사용된 은닉노드들의 활성화 함수는 sigmoid 함수 ($g(x)=1/(1+e^{-x})$)를 사용하였다. 모든 실험은 MATLAB으로 프로그램 하였고, 인텔 2.4GHz CPU, 2G RAM의 환경에서 실험되었다. 본 논문에서는 성능평가 기준으로 RMSE (Root Mean Squared Error)을 사용하였으며, 각각의 벤치마크 데이터에 대하여 20번의 실험을 통하여 평균을 구하였다.

표 1. 벤치마크 데이터의 사양
Table 1. Specification of benchmarking datasets

Name	No. of observations		Attributes
	Training data	Testing data	
Abalone	2000	2177	8
Ailerons	7154	6596	39
Auto price	80	79	15
Bank	4500	3692	8
Census (House8L)	10000	12784	8
Computer activity	4000	4192	10
Delta ailerons	3000	4129	5
Delta elevators	4000	5517	6
Kinematics	4000	4192	8
Puma	4500	3692	8

그림 3은 Delta ailerons을 이용한 실험에서 은닉노드 개수의 증가에 따른 FI-ELM의 학습, 검증 및 테스트 RMSE의 평균들을 나타냈다. 그림 3에서 훈련데이터의 RMSE는 은닉노드 개수의 증가에 따라 단계적으로 감소하며 검증데이터의 RMSE도 동일한 특성을 나타냄을 알 수 있다. 특히, 검증데이터를 이용하여 사전에 추가되어지는 은닉노드들의 학습 정도를 예측함으로써 테스트 데이터의 과잉학습에 따른 영향이 작음을 확인 할 수 있다.

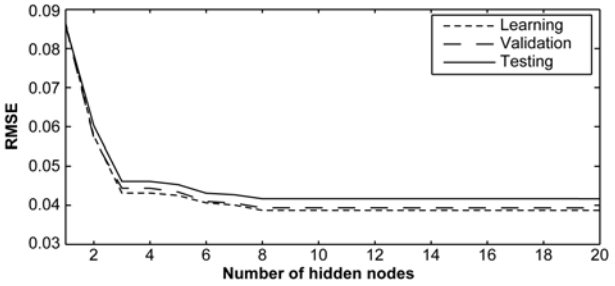

그림 3. FI-ELM의 은닉노드 개수의 증가에 따른 RMSE 변화 (Delta ailerons)
Fig. 3. RMSE curves of the FI-ELM with the number of hidden nodes to be added

그림 4는 Abalone을 이용한 실험에서 I-ELM과 FI-ELM의 은닉노드 개수의 증가에 따른 특성들을 보여준다. FI-ELM이 은닉노드 개수의 증가에 따라 RMSE가 I-ELM에 비하여 급격히 감소한다는 것을 알 수 있다. 그림 4는 1000개의 은닉노드들을 가진 I-ELM보다 약 10개 정도의 적은 수의 은닉노드들을 가진 FI-ELM이 더욱 우수한 성능을 가짐을 보여준다. 그림 5는 Abalone에 대하여 I-ELM과 FI-ELM의 은닉노드 개수의 증가에 따른 테스트 RMSE를 나타내었다. FI-ELM은 약 10개의 은닉노드들에서 빠른 수렴특성을 보여주며 테스트 RMSE가 0.0767인 반면에 I-ELM의 경우 0.1211로 제안된 방법이 적은 수의 은닉노드들에도 불구하고 기존의 방법에 비하여 월등히 우수한 성능을 가짐을 알 수 있다.

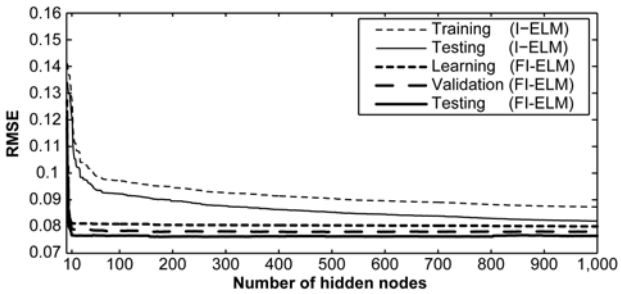

그림 4. 은닉노드 개수의 증가에 따른 I-ELM과 FI-ELM의 성능 비교 (Abalone)
Fig. 4. Performance comparison between I-ELM and FI-ELM with the number of hidden nodes

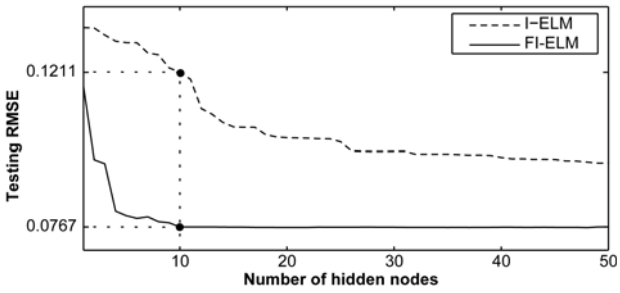

그림 5. 은닉노드 개수의 증가에 따른 I-ELM과 FI-ELM의 테스트 RMSE 비교 (Abalone)
Fig. 5. Testing RMSE performance comparison between I-ELM and FI-ELM with the number of hidden nodes

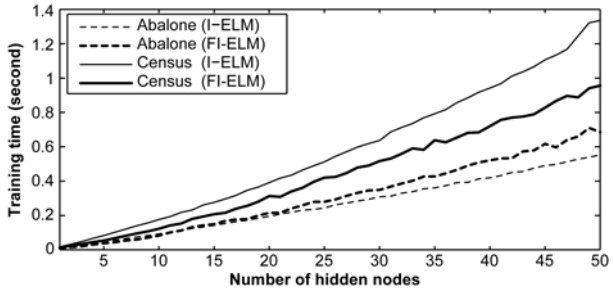

그림 6. I-ELM과 FI-ELM의 신경망 구축에 소요된 시간 비교 (Abalone)
Fig. 6. Training time comparison between I-ELM and FI-ELM

그림 6은 2000개의 훈련데이터를 가진 Abalone과 10000개의 훈련데이터를 가진 Census에서 은닉노드 개수에 따른 신경망 구축에 소요된 시간을 보여준다. FI-ELM은 I-ELM에 비하여 검증데이터에 대한 추가적인 계산이 필요하지만, 훈련 데이터를 학습과 검증데이터로 각각 50%씩 나눔으로써 데이터의 수가 I-ELM의 1/2이기 때문에 연산시간을 단축하는 효과를 얻을 수 있다. 특히 많은 수의 데이터에서는 I-ELM보다 유용함을 알 수 있다. 이것은 데이터의 개수가 많은 경우 출력 가중치들을 계산하기 위한 ELM의 역행렬 연산에 효율적임을 알 수 있다. 표 2는 최대 은닉노드 개수가 50개인 경우에 각각의 벤치마크 데이터들에 대한 RMSE의 평균과 표준편차를 나타냈다. 실험에 사용된 벤치마크 데이터들에 대하여 학습과 테스트 결과들은 제안된 FI-ELM이 동일한 개수의 은닉노드들에서 I-ELM보다 우수한 성능을 얻을 수 있음을 보여주며, 특히 테스트 데이터에서 FI-ELM은 I-ELM보다 더욱 작은 RMSE 표준편차를 얻을 수 있어 안정적인 성능을 얻을 수 있다. 표 3은 신경망의 최대 은닉노드 개수를 50개로 한 경우에 각각의 벤치마크 데이터들에 따른 신경망 구축에 소요된 시간을 상호 비교하여 나타냈다.

표 2. I-ELM과 FI-ELM 성능비교 (50개의 은닉노드인 경우)
Table 2. Performance comparison between I-ELM and FI-ELM (both with 50 hidden nodes)

Name	I-ELM		FI-ELM		
	Training	Testing	Learning	Validation	Testing
Abalone	0.1007±0.0056	0.0952±0.0051	0.0824±0.0014	0.0793±0.0012	**0.0769±0.0014**
Ailerons	0.1221±0.0525	0.1207±0.0508	0.0516±0.0029	0.0502±0.0028	**0.0511±0.0027**
Auto price	0.1060±0.0108	0.1083±0.0270	0.0941±0.0125	0.1003±0.0141	**0.0987±0.0234**
Bank	0.1677±0.0068	0.1694±0.0065	0.0691±0.0106	0.0669±0.0100	**0.0675±0.0105**
Census (House8L)	0.1008±0.0024	0.0978±0.0024	0.0775±0.0021	0.0858±0.0020	**0.0800±0.0017**
Computer activity	0.1802±0.0221	0.1740±0.0220	0.1366±0.0048	0.1333±0.0042	**0.1343±0.0058**
Delta ailerons	0.0742±0.0207	0.0782±0.0207	0.0384±4.5e-4	0.0388±3.5e-4	**0.0412±4.6e-4**
Delta elevators	0.0924±0.0136	0.0925±0.0139	0.0550±5.2e-4	0.0539±4.2e-4	**0.0533±4.4e-4**
Kinematics	0.1765±0.0166	0.1748±0.0150	0.1391±0.0030	0.1398±0.0028	**0.1435±0.0031**
Puma	0.2119±0.0139	0.2126±0.0141	0.1811±0.0024	0.1865±0.0025	**0.1849±0.0019**

표 3. I-ELM과 FI-ELM의 모델구축에 소요된 시간 비교 (50개의 은닉노드인 경우)
Table 3. Training time comparison between I-ELM and FI-ELM (both with 50 hidden nodes)

Name	I-ELM		FI-ELM		
	No. of training data	Time (s)	No. of learning data	No. of validation data	Time (s)
Abalone	2000	0.5529±0.0016	1000	1000	0.6698±0.0581
Ailerons	7154	1.4109±0.0201	3577	3577	1.0637±0.0843
Auto price	80	0.3946±0.0414	40	40	0.5724±0.0481
Bank	4500	0.7561±0.0020	2250	2250	0.5862±0.0391
Census (House8L)	10000	1.3355±0.0454	5000	5000	0.9559±0.0892
Computer activity	4000	0.7123±0.0127	2000	2000	0.6353±0.0424
Delta ailerons	3000	0.6510±0.0039	1500	1500	0.6940±0.0574
Delta elevators	4000	0.6905±0.0086	2000	2000	0.7779±0.0554
Kinematics	4000	0.6845±0.0050	2000	2000	0.6495±0.0676
Puma	4500	0.7025±0.0054	2250	2250	0.7688±0.0888

5. 결 론

본 논문에서는 flexible incremental 알고리즘을 이용하여 단계적으로 예측 잔류오차를 최소화하기 위하여 추가되어지는 은닉노드 개수를 신축성 있게 조절하고, 빠른 학습을 위하여 ELM을 이용하는 방법 (FI-ELM)을 제안하였다. 다양한 종류의 벤치마크 데이터들을 이용한 실험을 통하여 FI-ELM이 복잡한 비선형 회귀문제들에서 우수한 신경망을 구축할 수 있음을 확인하였다. FI-ELM은 신경망의 구축과정에서 사용자의 관여 없이도 자동적으로 최적의 신경망을 구축한다. 구축된 신경망 모델은 적은 개수의 은닉노드들에서도 기존의 방법에 비하여 우수한 성능을 얻을 수 있으며, 과잉학습의 영향이 적고 범용 근사화가 가능함을 보여주었다. 본 논문에서 제안된 방법은 복잡한 실질적인 문제들에서 신경망을 적용하는 방법으로 매우 유용하게 사용될 수 있으리라 예상된다. 아울러, Huang과 Chen은 최근 새로운 은닉노드를 추가할 때 기존의 존재하는 모든 출력 가중치들을 다시 갱신하는 CI-ELM (Convex I-ELM)과 다수의 후보 노드들을 생성한 후 최적의 노드를 선택하는 EI-ELM (Enhanced I-ELM)을 제안하였다 [10][11]. 본 논문에서 제안한 방법과 융합함으로써 더욱 우수한 성능을 얻을 수 있으리라 예상된다.

참 고 문 헌

[1] T. Andersen and T. Martinez, "Cross Validation and MLP Architecture Selection," *in Proc. of IEEE Int. Joint Conf. on Neural Networks IJCNN'99, CD Paper #192*, 1999.

[2] G. Castellano, A. M. Fanelli, and M. Pelillo, "An Iterative Pruning Algorithm for Feedforward Neural Networks," *IEEE Trans. on Neural Networks*, Vol. 8, No. 3, pp. 519-531, 1997.

[3] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," *in Proc. Int. Joint Conf. Neural Networks(IJCNN2004)*, Vol. 2, pp. 985-990. 2004.

[4] F. Han and D. S. Huang, "Improved Extreme Learning Machine for Function Approximation by Encoding a Priori Information," *Neurocomputing*, Vol. 69, No. 16-18, pp. 2369-2373, 2006.

[5] N. Liang, G. Huang, P. Saratchandran, and N. Sundararajan, "A Fast and Accurate Online Sequential Learning Algorithm for Feedforward Networks," *IEEE Trans. on Neural Networks*, Vol. 17, No. 6, pp. 1411-1423, 2006.

[6] H. T. Huynh and Y. Won, "Small Number of Hidden Units for ELM with Two-stage Linear Model," *IEICE Trans. on Information and Systems*, Vol. E91-D, No. 4, pp. 1042-1049, 2008.

[7] 조재훈, 이대종, 전명근, "Bacterial Foraging Algorithm을 이용한 Extreme Learning Machine의 파라미터 최적화," *한국지능시스템학회 논문지*, Vol. 17, No. 6, pp. 807-812, 2007.

[8] G. B. Huang, L. Chen, and C. Siew, "Universal Approximation Using Incremental Constructive Feedforward Networks With Random Hidden Nodes," *IEEE Trans. on Neural Networks*, Vol. 17, No. 4, pp. 879-892, 2006.

[9] C. Blake, C. Merz, UCI repository of machine learning databases, (http://www.ics.uci.edu/~mlearn/MLRepository. html), Department of Information and Computer Sciences, University of California, Irvine, USA, 1998.

[10] G. B. Huang and L. Chen, "Convex Incremental Extreme Learning Machine," *Neurocomputing*, Vol. 70, pp. 3056-3062, 2007.

[11] G. B. Huang and L. Chen, "Enhanced Random Search based Incremental Extreme Learning Machine," *Neurocomputing*, Vol. 71, pp. 3460-3468, 2008.

저 자 소 개

박진일(Jin-Il Park)

2001년 : 한밭대학교 제어계측공학과(학사)
2003년 : 한밭대학교 제어계측공학과 (공학석사)
2005년~현재 : 충북대학교 제어계측공학과 박사과정

관심분야 : 지능시스템, 다중생체인식, 퍼지이론, 패턴분류
E-mail : moralskr@yahoo.co.kr

정지석(Ji-Suk Jung)

2008년 : 충주대학교 전자공학과(학사)
2008년~현재 : 충북대학교 제어계측공학과 석사과정

관심분야 : 임베디드 시스템, 로봇공학, 지능시스템, 퍼지이론
E-mail : jsiscool@empal.com

조영임(Young-Im Cho)

2009년 제19권 3호 참조

전명근(Myung-Geun Chun)

2009년 제19권 3호 참조