2001년도 대한전기학회/대한전자공학회 시스템및제어분야 합동 추계학술대회 논문집 (2001.11.24)

진화신경망을 이용한 태풍 예측 시스템에 대한 연구

신대진+, 강환일*, 김갑일*

+인피니티(주), *명지대 전기정보제어공학부

A Study on the Typoon Prediction System Using the Evolving Neural network

+Dae Jin Shin, *Hwan Il Kang, *Kab Il Kim

+Infinity Inc., *Division of Electrical, Information & Control Eng., Myongji Univ.

Abstract - 본 논문에서는 태풍의 진로와 세기를 ES_BLRNN을 이용해 예측하였다. 기존의 방법인 수치모델이나, CLIPER모델을 사용함에 있어서, 통계적 방법인 CLIPER모델은 예측성능면에서 수치모델보다 그 성능이 떨어지고, 반면에 수치모델의 성능은 CLIPER 모델에 비해 우수하나 슈퍼컴퓨터(Cray-2S, FUSITSU)를 이용하여야만 예보가 가능한 제약점을 가지고 있다. 또한 수치모델을 슈퍼컴퓨터로 계산할 경우 약 30분 정도가 소요되는 점을 감안할 때, ES_BLRNN은 이들의 단점을 보안할 수 있는 하나의 방편이라 생각된다. 게다가 ES_BLRNN의 경우 개인용 컴퓨터로도 충분히 사용 가능할 만큼 비용이 저렴하고, 681개의 태풍을 학습할 때 걸리는 시간은 약 5분 정도이며, 146개의 태풍을 예측하는데 걸리는 시간은 약 3초 정도(Pentium MMX 200 Processor, RAM 64m, OS: RedHat LINUX 5.2, language : ANSI-C)로써, 슈퍼컴퓨터나 CLIPER모델에 비해 훨씬 빠르게 결과를 볼 수 있다.

1. 서 론

태풍의 진로및 중심기압을 예측하기 위해서는 태풍데이터들을 시계열(Time Series)[1]로 변환하여야 한다. 시계열은 정해진 순간마다 측정된 값($x_1, x_2, x_3, \ldots, x_n$)들의 모임[1]을 말하며, 시계열 예측은 사업계획수립, 일기예보, 주가예측과 다양한 신호처리분야 등에서 활발히 응용발전하고 있다. 일반적으로 시계열은 혼돈된 신호이거나 불규칙적인 특성을 가지고 있어 이들을 함수로 모델링 하기에는 부족함이 있다. 따라서 근래에는 시계열에서 퍼지규칙과 적당한 소속함수를 선택하여 시계열을 모델링 하려는 경향이 있다.[2]

본 논문에서는 태풍의 진로와 중심기압을 시계열화 하고, 그 분석을 위해 신경망과 진화전략을 사용하였다. 여기에 사용된 신경망 학습 알고리즘은 쌍선형 회귀성 신경망(BLRNN)[3]이며, 이를 좀더 보완하고자 BLRNN에 진화전략[4]을 적용하였다. 입력 패턴으로는 6시간 간격의 태풍의 위치(위도, 경도), 중심최저기압 및 날짜를 사용하였으며, 12시간 후의 태풍의 위치와 중심최저기압을 출력 패턴으로 사용하였다. 2장에서는 BLRNN과 진화전략및 진화전략을 이용한 쌍선형 회귀성 신경망(ES_BLRNN)에 대한 설명, 4장에서는 데이터의 수집과, 가공, 학습 및 학습 결과에 대하여 설명하고, 5장에서는 예측결과를 나타내었다.

2. BLRNN과 진화전략

2.1 BLRNN

그림 1는 입력변수 3개, 은닉층 뉴런2개, 2차의 회귀도, 그리고 한개의 출력 변수를 가지는 BLRNN이 보여지고 있다.

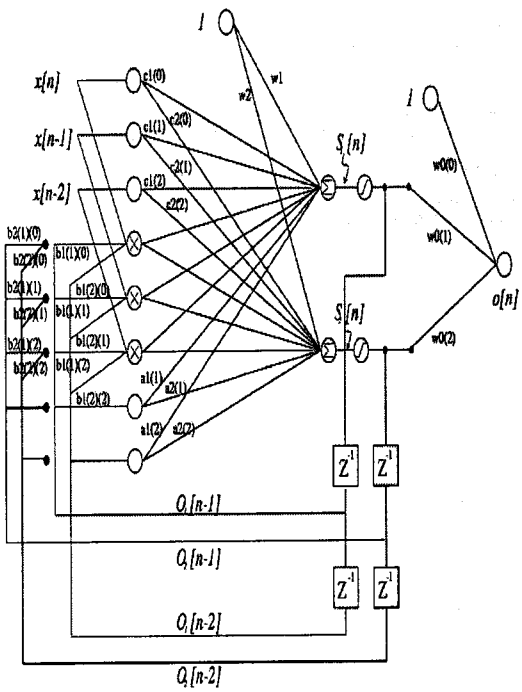

그림 1. 3-2-1의 구조를 갖는 BLRNN의 예 (N=3, M=2, K=2)

그림 1에서 입력 vector를 다음과 같이 정의 하면,

$\vec{X}[n]=[x[n],\ x[n-1],\ \cdots,\ x[n-K]]^T$
이고, M차원의 출력 vector가 $\vec{o}[n]$이므로

$\vec{o}[n]=[o_1[n],\ o_2[n],\ \cdots,\ o_M[n]]^T$
이 된다. 그러므로 회귀항은 M by K(=N-1) 의 형태로 나타낼 수 있다. 즉,

$$\vec{Z}[n]=\begin{pmatrix} o_1[n-1] & o_1[n-2] & \cdots o_1[n-K] \\ o_2[n-1] & o_2[n-2] & \cdots o_2[n-K] \\ \vdots & \vdots & \ddots \ \vdots \\ o_M[n-1] & o_M[n-2] & \cdots o_M[n-K] \end{pmatrix}$$

$\vec{Z_p}[n]=[o_p[n-1], o_p[n-2], \cdots, o_p[n-K]]$, (p = 1, 2, ···, M)
이다. 또한, 각 은닉 뉴런에서의 합을 $S_p[n]$이라 정의하면, 위 식의 M by K 회귀항 정리를 이용해 다음과 같이 표현할 수 있다.

$$\begin{aligned} S_p[n] &= w_p + \sum_{k_1=1}^{N-1} a_p[k_1] o_p[n-k_1] \\ &+ \sum_{k_1=1}^{N-1}\sum_{k_2=0}^{N-1} b_p[k_1,k_2]\ o_p[n-k_1]\ x[n-k_2] \\ &+ \sum_{k_2=0}^{N-1} c_p[k_2] x[n-k_2] \end{aligned}$$

$$= w_p + \vec{A_p}\,\vec{Z_p}^T[n] + \vec{Z_p}[n] B_p \vec{X}[n] + \vec{C_p}\,\vec{X}[n] \quad (3)$$

이때 $\vec{A_p}$는 회귀항의 weight vector이며, B_p는 회귀항과 feedforward층의 weight matrix, $\vec{C_p}$는 feedforward층의 weight vector 이다. 여기에 활성화 함수(φ)을 고려한 마지막 출력, 즉 $o[n]$은 다음과 같이 정의 할 수 있다.[3]

$$o[n]=w_o+\sum_{p=1}^{M}\varphi(S_p[n])w_{op} \quad (1)$$

w_o는 최종 출력단의 BIAS 이고, w_{op}는 은닉층의 각각 출력에 해당하는 weight이다. 즉, BLRNN에서의 은닉층과 출력층 사이 연결은 보통의 feedforward형 신경망에서 쓰이는 관계식과 같다[3]. 다음으로 BLRNN의 학습 알고리즘에 대해 살펴보자. 이는 기본적으로 gradient descent 방법에 의해 구해지며, error를 다음과 같이 정의하면,

$$E[n]=\frac{1}{2}\sum_{l=1}^{N_o}(d_l[n]-o_l[n])^2 \quad (2)$$

여기서 N_o는 출력층 뉴런의 수이며, $d_l[n]$은 l번째 출력층 데이터에 대한 목표값(target)이다. 마지막으로 gradient descent 방법에 의한 weight 조정은 다음 식으로 표현된다.

$$w_{ij}^{new}=w_{ij}^{old}-\eta\frac{\partial E[n]}{\partial w_{ij}^{old}} \quad (3)$$

여기서 η는 학습 이득이고, w_{ij}는 j에서 i사이의 weight를 나타낸다. 윗식의 적당한 미분값을 구하여 대입하면, 매 iteration마다의 weight 변화는 다음 식들로 요약된다.

$$w_{lp}^{new}=w_{lp}^{old}+\eta(w_{lp})\sum_{n=1}^{N_{frame}}\delta_l[n]o_p[n] \quad (4)$$

$$C_p^{new}=C_p^{old}+\eta(c_p)\sum_{n=1}^{N_{frame}}\delta_p[n]X[n] \quad (5)$$

$$A_p^{new}=A_p^{old}+\eta(a_p)\sum_{n=1}^{N_{frame}}\delta_p[n]Z[n] \quad (6)$$

$$B_p^{new}=B_p^{old}+\eta(b_p)\sum_{n=1}^{N_{frame}}\delta_p[n]Z[n]^TX[n] \quad (7)$$

여기서 $\delta[n]$은 delta rule[15,16,17]에 의해 구해진 weight의 변화율을 나타내며, $\delta_l[n]$는 BIAS의 weight를, $\delta_p[n]$는 입력 및 회귀항 weight의 변화율을 나타낸다. 또한 수렴 조건은 앞에서 정의한 식 (5)(Error함수) 의 미분치가 음수가 되기 위한 필요한 조건이므로 이들을 정리하면,

$$0 < \eta < \frac{2N_{frame}N_o}{\sum_{n=1}^{N_{frame}}\sum_{l=1}^{N_o} sup(\frac{\partial o_l[n]}{\partial w})^2} \quad (8)$$

이다. 각 η에 따른 수렴 조건은 [3]를 참조하면 된다.

2.2 진화 전략(Evolution Strategy)

진화 전략은 매개변수 최적화 문제를 해결할 수 있는 방법으로써 자연 진화의 원리를 모방한 알고리즘이다.[4] 이것은 1960년대 독일의 Rechenberg[5]가 제안했고 Schwefel[6]에 의해 개선되었다. 초기의 진화 전략은 돌연변이 연산자만을 가지고 부동점 표현이 사용된 진화 프로그램[4,5,6]으로 인식될 수 있다. 그러나 최근에는 교배연산자도 돌연변이와 함께 쓰이고 있다. 초기 진화 전략은 연속적으로 변하는 매개변수들을 갖는 다양한 최적화 문제들에 적용해 왔으나 이산 문제들에 대해서 적용이 확장되었다.[4]

2.2.1 두회원 진화 전략

초기 진화 전략은 하나의 개체만으로 구성된 개체집단을 사용하였고, 또한 진화과정에서 단 하나의 유전 연산자(돌연변이)만을 사용였다. 즉, $v = (x, \sigma)$, 여기서 첫 번째 벡터 x는 탐색 공간내의 하나의 점을 나타낸다. 두 번째 벡터 σ는 표준 편차들의 벡터이다. 돌연변이는 x를 다음의 수식으로 표현함으로써 실현된다.[4]

$$x^{t+1} = x^t + N(0, \sigma) \quad (9)$$

여기서 $N(0, \sigma)$는 평균이 0이고 표준편차가 σ인 독립적인 랜덤 Gaussian 숫자이다. 자손세대가 만약 좀더 좋은 적합도를 갖고 모든 구속 조건들이 만족된다면 개체 집단의 부모세대와 자손세대의 교체가 일어나게 된다. 즉, 적합도 함수를 $f(x)$라 정의하면, 자손세대 (x^{t+1}, σ)는 $f(x^{t+1}) > f(x^t)$일 때만 그것들의 부모세대와 바뀐다. 만약 그렇지 않을 경우 자손세대는 제거되고 개체 집단은 변화 없이 남는다. 그러므로 경쟁단계에서는 자손세대와 부모세대가 같이 경쟁하므로 이를 두회원 진화전략이라고 부른다.[4]

2.2.2 다회원 진화 전략

다회원 진화전략은 이전의 두회원 진화 전략과 개체집단의 크기에서 다르다고 할 수 있다. 또한 다회원 진화 전략의 특징은 집단 내의 모든 개체들은 같은 교배 확률

을 갖고 교배 연산자에 의해 교배한다.[4]
즉, 두 부모세대를 다음과 같이 정의하면,

$$(x^1 , \sigma^1) = ((x_1^1 \dots x_n^1),(\sigma_1^1 \dots \sigma_n^1)) \text{ and}$$

$$(x^2 , \sigma^2) = ((x_1^2 \dots x_n^2),(\sigma_1^2 \dots \sigma_n^2))$$

이고 이때의 자식세대는

$$(x , \sigma) = ((x_1^{q_1} \dots x_n^{q_n}),(\sigma_1^{q_1} \dots \sigma_n^{q_n}))$$

이 된다. 여기서 $q_i = 1$ 또는 $q_i = 2$ 이고 모든 $i = 1, \dots, n$에 대해서 동등한 확률을 갖는다. 이러한 연산자들은 전역적인 형태로 적용될 수 있는데, 새로운 부모 세대들의 쌍은 자손세대 벡터의 각 원소를 위해서 선택되어 진다. 또한, 돌연변이를 통해 생성된 새로운 자손세대를 (x' , σ')로 표현할 수 있다. 즉,

$$\sigma' = \sigma \cdot e^{N(0, \Delta\sigma)} \tag{10}$$

$$x' = x + N(o, \sigma') \tag{11}$$

이고, 여기서 $\Delta\sigma$는 매개 변수가 된다.[4]
진화 전략은 실수 함수 최적화 문제들을 위해서 만들어졌기 때문에, 실수 수치 문제 영역 안에서 매우 좋은 성능을 나타낸다.[4]

3. Network 모델링

본문에서는 BLRNN중 sigmoid의 기울기를 결정하는데 위의 다회원 진화 전략이 적용되었으며, 그 절차는 다음과 같다.

while desired Error or Maximum epoch
BLRNN;
if Error >= previous Error /*start Evolution Strategy*/
save α, β ;
/* denotes slope of activation function in hidden layer */
/* denotes slope of activation function in output layer */
determine σ;
determine initial $\Delta\sigma$
while number_of_ES <= Maximum_number_of_ES
create new Generation with $\Delta\sigma$
create new , with new Generation;
BLRNN with new α, β, ;
calculate new_Error;
if Error>new_Error
modify into new ;
modify into new ;
send stop condition to while statement
end if
else
update $\Delta\sigma$
end else
number_of_ES++;
end while
if number_of_ES = Maximum_number_of_ES
Load saved α, β, ;
send stop condition to while statement
end if
end if
epoch++;
end while

4. 원시 태풍 데이터의 가공

신경망의 더 우수한 출력을 위해서 각 데이터들을 가공할 필요가 있다.[7] 각각의 데이터는 편차와 평균치가 서로 다르므로 신경망을 적용할 경우 평균치가 큰 데이터에 가중치가 많이 적용되어진다. 그러므로 각 데이터를 0에 가까운 값에서부터 1에 가까운 값으로 mapping하여 그 편차 및 평균치를 거의 동일하게 적용, 데이터 상호간에 그 편차 혹은 평균치로 인한 잘못된 학습이 진행됨을 막을 수 있다. 이는 데이터의 정규화(normalize)라 불리며, 신경망에서 가장 중요한 절차 중 하나라고 말할 수 있다.[7] 본 논문에서 사용된 데이터중 위도의 경우를 살펴보면, 위도의 데이터는 0 °부터 90 °까지의 분포하며, 각 태풍은 그 생성된 위치 및 소멸된 위치가 서로 다르다. 따라서 만약 위도성분 데이터를 정규화 없이 입력패턴으로 사용한다면, 적도부근에서 발생된 태풍 즉, 위도성분이 0에 가까운 태풍들은 그 가중치가 작아지며 적도에서 떨어져 발생한 태풍일수록 입력이 20에서 60정도의 실수가 사용되게 되므로 활성화 함수(sigmoid 함수 적용)의 출력은 항상 1이 된다. 그러므로 절절한 학습이 진행되지 않으며, 또한 진행하고자 할 경우에는 보다 복잡한 초기 가중치들의 선정과 보다 정밀한 학습율 및 선형전이함수의 기울기가 요구된다. 따라서, 본 논문에서 사용된 각 데이터는 0.1에서 0.9까지로 mapping하였다.

4.1 학습

학습을 위한 네트워크는 위도 예측을 위한 네트워크, 경도 예측을 위한 네트워크, 중심기압 예측을 위한 네트워크 등 총 3개의 네트워크로 구성한다. 각 네트워크에 요구되는 매개변수들은 실험을 통하여 결정하였고, 선형전이 함수의 기울기는 진화전략을 이용해서 찾게된다. 또한 초기 가중치는(Initial Weight) 0에서 1사이의 랜덤한 값들로 정하고, 입력패턴은 Sliding window 기법[8]을 이용하며, $x(t), x(t-1), x(t-2)$로 정하였고, 출력 패턴으로는 $x(t+2)$로 정하였다. 이는 12시간 후 태풍의 위치 예측을 의미한다. 태풍진로예측은 태풍의 전향점[8]을 기준으로 다른 네트워크를 사용하던 기존 방법과는 달리 어떤 태풍이더라도 12시간 예측을 가능하게 하기 위해 하나의 네트워크로써 그 전향여부에 상관없이 학습하였으며, 또한 각 태풍을 유형별로 나누지 않고, 태풍 발생 후 6 step (36시간)만에 소멸된 태풍을 제외하고는 모든 태풍을 학습 입력으로 사용한다. 그 결과의 한 예로 태풍의 경도를 학습한 결과를 다음 그림 2에 도시하였다.

5. 예측결과

예측시 사용된 네트워크는 학습시 사용한 네트워크와 동일하며, 학습의 결과로 얻어지는 최종 가중치를 전달받아 예측하게 된다. 예측으로 사용된 데이터는 1991년 1월 1일부터 1996년 12월 31일 까지 총 146개의 태풍 4,455개의 패턴을 사용하고, 12시간 예측을 위해 발생 후 6step (36시간)안에 소멸한 태풍을 제외하였

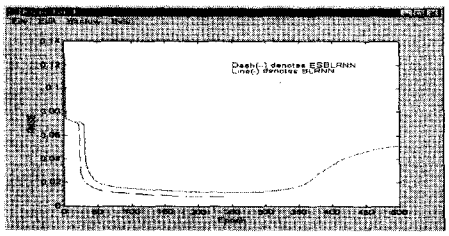

그림 2. ES_BLRNN과 BLRNN의 학습 성능 비교(RMSE)

다. 표 1에서는 예측에 사용된 모든 태풍데이터를 예측한 결과이다.

표 1. 91년부터 96년까지 총 146개의 태풍 예측치에 대한 RMSE

네트워크 / Error	위도	경도	중심기압
RMSE	0.005975	0.008517	0.011183

ES_BLRNN 성능을 구체적으로 설명하기 위하여 그림 3에 예측된 데이터중 임의 2개 태풍의 위도, 경도, 중심기압을 3차원으로 나타낸다.

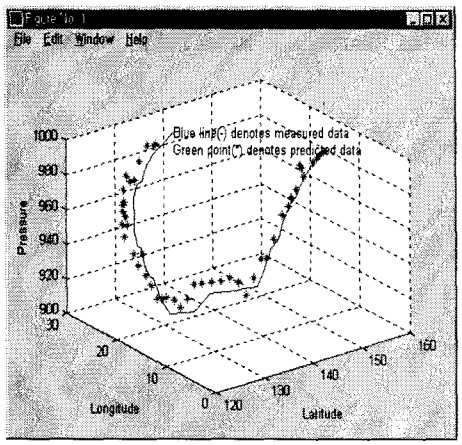

그림 3. 위도, 경도, 중심기압의 측정치와 예측치의 비교 (91년 5월 7일 0시 발생, 5월 17일 0시 소멸한 태풍)

그림 3에서 보여지듯 태풍의 데이터는 그 비선형성이 매우 크다고 할 수 있으나, ES_BLRNN을 이용하여 위도, 경도및 중심기압을 비교적 정확하게 예측할 수 있다.

6. 결 론

본 논문에서는 태풍의 진로와 세기를 ES_BLRNN을 이용해 예측하였다. 기존의 방법인 수치모델이나, CLIPER모델을 사용함에 있어서, 통계적 방법인 CLIPER모델은 예측성능면에서 수치모델보다 그 성능이 떨어지고, 반면에 수치모델은 성능은 CLIPER모델에 비해 우수하나 슈퍼컴퓨터(Cray-2S, FUSITSU)를 이용하여야만 예보가 가능한 제약점을 가지고 있다. 또한 수치모델을 슈퍼컴퓨터로 계산할 경우 약 30분 정도가 소요되는 점을 감안할 때, ES_BLRNN은 이들의 단점을 보안할 수 있는 하나의 방편이라 생각된다. 게다가 ES_BLRNN의 경우 개인용 컴퓨터로도 충분히 사용 가능할 만큼 비용이 저렴하고, 681개의 태풍을 학습할 때 걸리는 시간은 약 5분 정도이며, 146개의 태풍을 예측하는데 걸리는 시간은 약 3초 정도(Pentium MMX 200 Processor, RAM 64m, OS: RedHat LINUX 5.2, language : ANSI-C)로써, 슈퍼컴퓨터나 CLIPER모델에 비해 훨씬 빠르게 결과를 볼 수 있다.

[참 고 문 헌]

[1] J. R. Jang & C. Sun, "Prediction Chaotic Time series with Fuzzy If_Then rules," Second IEEE International Conference on Fuzzy Systems, San Francisco, pp. 1079-1084, 1993.

[2] R. M. Tong, "The evaluation of Fuzzy Models derived from Experimental Data," Fuzzy sets and Systems, vol. 4, pp. 1-12, 1980.

[3] Dong C. Park and Yan Zhu, "Bilinear Recurrent Neural Network," IEEE ICNN, vol.3, pp. 1459-1464, 1994.

[4] Zbigniwe Michalewicz, *Genetic Algorithms + Data Structures=Evolution Programs*, Springer Verlag, 1994. pp. 167-184.

[5] I. Rechenbeg, "Cyvernetic Sloution Path of an Experimental problem", Royal Aircraft Establishment, Library Translation

[6] H.-P. Schwefel, "Kybernetiche Evolution als Strategie der Experimentellen Forshung in der Stromungstechmik", Diplomarveit, Techmishe Universitat, Berlin, 1965.

[7] K. Hornik, M. Stinchcombe and H. White, "Multilayer feedback networks are universal approximators," Neural Network, vol. 2, pp. 359-366, 1989.

[8] G.Johnson and F.Lin "Hurricane Tracking via Backpropagation Neural Network," Proceeding of IEEE International Conference on Neural Networks, pp. 1103-1106, 1995.