2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

뒤틀림 현상이 없는 FSOM 학습 알고리즘

정선정*, 정순호**
*부경대학교 전자계산학과
**부경대학교 전자컴퓨터정보통신 공학부
e-mail:sjjung@mail1.pknu.ac.kr

Improved Fast SOM learning algorithm without cross-over

Sun-Jung Jung*, Soon-Ho Jung**
*Dept of Computer Science, Pukyong University
**Faculty of Electronics, Computer, Telecommunication Engineering Pukyong National University

요약

자기구성 특징지도(Self-Organizing feature Map : SOM) 및 L^* 등의 자가 학습 신경망의 알고리즘들은 학습 결과 중에 바람직하지 못한 뒤틀림 현상(cross-over)을 생성하게 되므로 재학습으로 인한 전반적인 학습 시간의 지연을 초래한다. 이 논문에서는 비교적 학습 속도가 빠른 L^* 의 점증적 학습 구조를 기본으로 하여 뒤틀림 현상 방지를 목적으로 초기 학습 단계에서 학습 가중치들의 노드들을 재조정하는 개선된 알고리즘을 제안한다. 이러한 알고리즘의 실험 결과는 모두 정상적인 학습 결과를 보이고 학습의 시행 착오적인 재실행이 없으므로 전반적인 학습 속도는 기존의 알고리즘보다 빠르게 됨을 보인다.

1. 서론

Kohonen이 제안한 Self-Organizing feature Map(SOM)은 경쟁적으로 자가 학습하는 신경망의 한 구조이다[1]. 이 SOM의 학습 시간을 단축하기 위한 방법으로 단계별로 학습하는 L^* 알고리즘이 제안되었다[2]. 이러한 신경망들은 바람직하지 못한 학습 결과로 약 50%의 뒤틀림 현상(cross-over)을 생성하게 된다. 이러한 잘못된 결과를 방지하기 위한 방안이 제시되었고 이 방법이 일반적 SOM에 적용되었다[3].

본 논문에서는 학습 속도가 빠른 L^* 알고리즘의 뒤틀림 현상을 제거하기 위하여 구조를 변경하고 일부 학습 단계의 재구성을 통하여 이 현상을 방지하는 알고리즘을 제안한다.

이 논문은 2장에서 SOM의 학습 방법과 L^* 알고리즘을 이용한 SOM의 학습 방법을 소개하고, 3장에서 제안하는 알고리즘을 기술하였고 4장에서는 제안한 알고리즘을 평가하기 위해 단계별 학습횟수와 map의 크기에 따른 수행 속도를 실험하고 분석하고 5장에서는 결론을 언급한다.

2. 관련 연구

2.1 Self-Organizing feature Map

SOM은 입력층과 경쟁층으로 이루어져 경쟁 학습을 하는 신경망 구조 중 하나이며, 이때 경쟁층의 뉴런들은 완전 연결되어 있다[4][5].

SOM의 학습 방법을 살펴보면, 먼저 입력층에 입력이 들어오면 입력과 경쟁층의 뉴런과의 거리를 구하는데, 이때 거리가 가장 작은 뉴런을 승리자라 부른다. 승리자를 구하는 방법은 식 (1)과 같다[6][7].

$$||E - U_c|| = \min ||E - U_i||. \qquad \text{식(1)}$$

여기서 E는 입력, U_c는 승리자를 의미한다.

그런 다음 앞에서 선택한 승리자를 중심으로 미

리 정해진 이웃의 반경에 속하는 뉴런들의 가중치를 수정한다. 이때 이웃의 반경은 N_c이며 학습이 진행됨에 따라 점차 줄어든다. 식(2)는 가중치를 수정하는 식을 나타낸다.

$$\Delta u_{ij} = \begin{cases} \alpha(e_j - u_{ij}), & i \in N_c \\ 0, & otherwise \end{cases}$$

$$u_{ij}^{new} = u_{ij}^{old} + \Delta u_{ij} . \qquad 식(2)$$

여기서 α는 학습률을 의미한다.

2.2 L* 알고리즘

L* 알고리즘은 SOM의 학습 시간을 줄이기 위해 제안된 알고리즘으로 학습을 단계별로 나누어서 수행하는 방법이다[8]. 학습을 할 때 모든 뉴런을 한꺼번에 학습을 하는 것이 아니라 점차적으로 뉴런의 수를 증가시키면서 학습하는 방법이다.

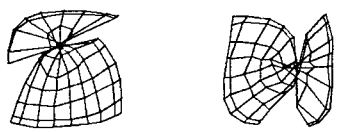
L* 알고리즘의 학습 과정을 간략히 살펴보면 먼저 첫 번째 단계에서는 일부의 뉴런들만을 SOM과 같은 방식으로 학습을 한다. 다음 단계에서는 첫 번째 단계에서 각각 학습된 뉴런에 일부의 뉴런을 추가해 승리자를 찾은 다음 뉴런들의 가중치를 수정한다.

이 알고리즘은 이전 단계에서 이미 정돈된 뉴런들로 인해 전체적인 학습 속도를 감소시키는 개선된 SOM 방법이다.

3. 개선된 L* 알고리즘

일반적 SOM이나 L* 알고리즘에서 50% 확률로 학습 결과의 뒤틀림 현상을 보게 된다. 뒤틀림 현상은 [그림 1]과 같이 각 뉴런의 위상적 위치가 뒤틀리는 현상으로 이것은 가중치의 초기 상태와 학습할 패턴들의 순서에 영향을 받는다[1].

이러한 뒤틀림 현상으로 인해 기존의 방법에서는 학습을 처음부터 다시 해야 하고, 이는 학습시간을 증가시키는 작용을 하게 된다. 따라서 이러한 결과를 방지하기 위한 학습 구조와 알고리즘을 소개한다.

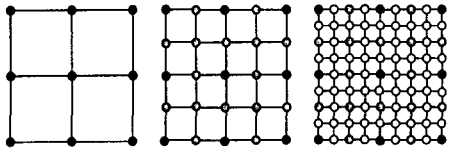


[그림 1] 뒤틀림 현상

3.1 학습 구조

3.1.1. 학습 뉴런수의 점증기법

신경망 학습에서 map의 크기에 따른 학습 단계수는 $\log(M-1)$으로 이때 M은 map의 크기이다. 각 i단계에서 $(2^i+1)^2$개의 학습 뉴런을 채택하고 k단계의 뉴런들은 k-1단계에서의 뉴런을 포함에서 [그림 2]와 같이 점증적으로 증가시킨다.



(a) 1단계 (b) 2단계 (c) 3단계

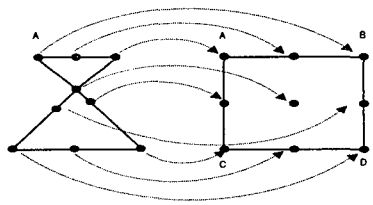
● : 1단계 학습에 참여하는 뉴런
◎ : 2단계 학습에 추가되는 뉴런
○ : 3단계 학습에 추가되는 뉴런

[그림 2] 점증 기법의 학습 뉴런

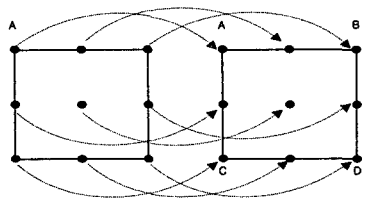
map의 크기가 9의 경우이므로 단계수는 3단계가 되고 각 단계별로 뉴런수는 9, 25, 81순으로 학습된다. 일반적으로 이 기법을 이용한 모든 학습은 9개의 뉴런으로 시작한다.

3.1.2. 뒤틀림 방지

이러한 점증 기법의 학습 구조에서 첫 번째 단계의 9개의 뉴런의 학습 결과를 보면 최종 결과의 뒤틀림 현상을 일찍이 감지할 수 있다. 따라서 첫 번째 단계의 학습 결과에서 각 뉴런의 값의 재조정을 [그림 3]과 같이 A 노드의 가중치 값으로부터 가장 먼 가중치 값을 D의 값으로 정하고 그 나머지를 B와 C로 정한다. 그런 뒤 그 노드들 사이의 가중치들도 인근 관계가 대응되는 노드에 할당한다.



(a) 뒤틀림 현상이 있는 경우



(b) 제대로 학습된 경우

[그림 3] 뉴런의 위치를 재조정하는 방법

3.2 개선된 알고리즘

뒤틀림 현상을 방지하는 개선된 L^* 알고리즘의 학습 알고리즘은 [그림 4]와 같다. 여기서 학습 단계 수는 $\log(M-1)$이고 이때 M은 map의 크기이다. 다음 첫번째 단계를 기존의 L^* 알고리즘과 같은 방법으로 학습을 수행하나 첫번째 단계를 수행한 후 뉴런의 가중치를 재조정한다.

뉴런의 재조정이 끝나면 이 재조정된 뉴런의 뒤틀림 방지를 위한 가중치를 가지고 다음 단계의 학습을 수행하는데, 다음 단계부터는 재조정 단계가 필요하지 않는다.

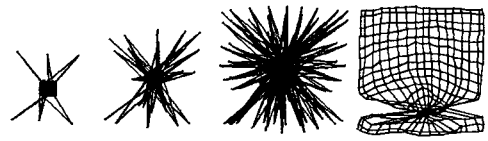
```
procedure 개선된 L*()
 for level=1 to log(M-1) by 1
  repeat ∧× ( 2^level+1)^2 // ∧ : 학습 횟수
   // winning unit 찾기
   for k<-1 to level by 1
     δ=(M−1)/ 2^k
    for x<-max(1, x_w−δ) to min( x_w+δ, M) by δ
    for y<-max(1, y_w−δ) to min( y_w+δ, M) by δ
      winner 선택
   alpha, neighborhood 설정
    δ=(M−1)/ 2^level
   // weight update
   for x<-max(1, x_w−δ) to min( x_w+δ, M) by δ
   for y<-max(1, y_w−δ) to min( y_w+δ, M) by δ
      u_ij^new = u_ij^old + Δ u_ij
  end repeat
  // 뉴런 재조정
   if(level==1)
    // 4개의 corner에 위치한 뉴런 재배치
    for x<-0 to M by δ
    for y<-0 to M by δ
      u_MM <- u_00로부터 최대 거리를 가진
              가중치를 할당
      u_0M, u_M0<- u_00으로부터 가중치 값 부여
    // 나머지 뉴런 재배치
    for x<-0 to M by δ
    for y<-0 to M by δ
      u_xy<-나머지 뉴런들의 가중치를 인근관계 고려
            해서 할당
   end if
  end for level
end procedure
```

[그림 4] 개선된 알고리즘

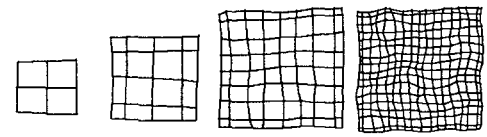
4. 실험

개선된 알고리즘을 평가하기 위해 다음과 같은 실험이 수행된다. 실험은 Pentium III 800 MHz의 IBM PC에서 실행되었고, 사용된 프로그래밍 언어는 Visual C++이다.

먼저 학습 결과를 비교하기 위해 map의 크기가 17일 때의 기존의 알고리즘과 개선된 알고리즘의 결과를 비교하면 그 결과는 [그림 5]와 같다. (a)는 뒤틀림 상태인 L^* 알고리즘의 단계별 학습 결과를 나타내는 것이고 (b)는 제안한 알고리즘의 단계별 학습 결과를 보여주는 것이다. 이때 1000개의 입력 패턴과 단계별 학습 횟수는 3000을 사용한다.



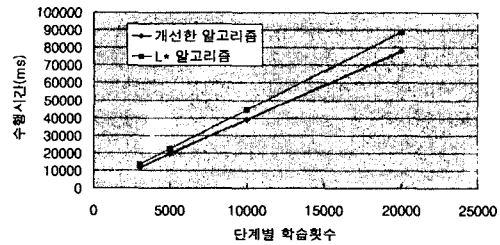
(a) 기존의 알고리즘의 단계별 학습 결과



(b) 개선된 알고리즘의 단계별 학습 결과

[그림 5] 학습 결과 비교

다음으로 정상적 결과를 생성한 경우에 기존의 알고리즘과 개선된 알고리즘의 학습 시간을 비교하면 단계별 학습횟수에 따른 두 알고리즘의 학습 시간을 비교 결과는 [그림 6]과 같고 이때 map의 크기는 17이고, 입력은 1000이다.

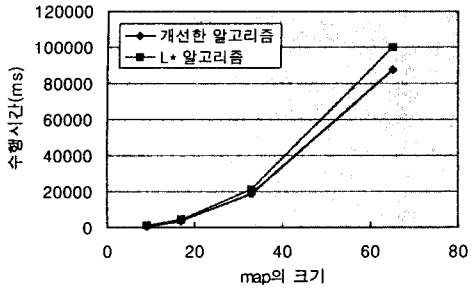


[그림 6] 학습횟수에 따른 학습시간 그래프

[그림 6]의 그래프에서도 알 수 있듯이 학습횟수가 커질수록 제안한 알고리즘이 L^* 알고리즘보다 수행 시간이 빠르며 평균적으로 제안한 알고리즘이 약 1.14배 빠르다는 것을 알 수 있다.

다음으로 정상적 결과를 나타내는 경우에 map의 크기에 따른 두 알고리즘의 수행 시간을 비교하기 위한 실험 결과는 [그림 7]과 같다. 이때 입력 패턴

은 1000, 단계별 학습횟수는 1000이다.



[그림 7] map의 크기에 따른 학습시간 그래프

map의 크기에 대한 두 알고리즘의 수행 속도 역시 제안한 알고리즘이 기존의 알고리즘보다 속도가 향상되었고, 평균적으로 수행속도가 약 1.13배 단축된다는 것을 알 수 있다.

앞에서 살펴본 두 가지 실험 결과는 뒤틀림 현상의 결과가 생기지 않을 경우의 제안한 알고리즘과 기존의 알고리즘을 비교한 결과이다. 따라서 이러한 뒤틀림 현상을 고려하여 비교하게 되면 기존의 알고리즘은 뒤틀림 결과가 생겼을 때 처음부터 다시 학습을 해야 하므로 앞에서 실험한 결과의 2배의 이상의 시간이 걸리게 되므로 총체적 학습 시간은 이 논문에서 제안한 알고리즘이 빠르다는 것을 보인다.

5. 결론

본 논문에서는 SOM의 학습 속도를 높이는 동시에 뒤틀림 결과를 방지한 개선된 알고리즘을 제안하였다. 즉 SOM의 학습 속도를 향상시키기 위해 기존의 L^* 알고리즘의 점증 기법의 특성을 이용하였고, 학습 구조를 기반으로 한 뉴런의 가중치들을 재조정하는 단계를 추가하는 학습 알고리즘을 제안한였다. 이 알고리즘에 따른 실험을 통해서 제안한 알고리즘이 전혀 뒤틀림 현상이 없으며 학습 시간 또한 기존의 알고리즘보다 훨씬 빠르다는 것을 알 수 있다.

참고문헌

[1] Judith E. Dayhoff, "Neural Network Architectures An Introduction", VAN NOSTRAND REINHOLD, 1990.

[2] Y.P.Jun, H.Yoon and J.W.Cho, "L* Learning: A Fast Self-Organizing Feature Map Learning Algorithm Based on Incremental Ordering", IEICE Trans. Inf. & Sys. vol. E76-D, No.6, June. 1993.

[3] Mu-Chun Ju, Hsiao-Te Chang, "Fast Self-Organizing Feature Map Algorithm", IEEE trans. on Neural Networks, voll. II, No.3, pp. 721-733, 2000.

[4] Koikkalainen, P., Oja, E., "Self-Organizing Hierarchical Feature Maps," Proc. IEEE International Conference on Neural Networks, vol. II. pp. 279-284, 1990.

[5] Fang, L., Jennings, A., Wen, W. X., Li, K. Q-Q., Li, T., "Unsupervised Learning for Neural Trees," Proc. IEEE International Joint Conference on Neural Networks, (Singapore), vol. III, pp. 2709-2715, Nov. 1991.

[6] Kohenen. T, "The self-organizing map, Proc. IEEE 78(9), 146-180, 1990.

[7] S.Haykin, "Neural Networks", Prentice Hall, 1994.

[8] Y.P.Jun, H.Yoon and J.W.Cho, "L* Learning: A Fast Self-Organizing Feature Map Learning Algorithm Based on Incremental Ordering", IEICE Trans. Inf. & Sys. vol. E76-D, No.6, June. 1993.