SOM을 이용한 LVQ 네트워크 설계

김영렬* · 이용구** · 손동설*** · 강성호* · 엄기환*

*동국대학교 전자공학과, **한림정보산업대학 전자통신, ***유한대학

LVQ Network Design using SOM

Young lyul Kim* · Yong-gu Lee** · Dong-seol Son*** · Seong Ho Kang* · Ki-hHwan Eom*

*Dept. of Electronic Eng, Dongguk University, Dept. of , **Electronic Communication, Hallym College of Information & Indust, ***Yuhan College

E-mail : kimyl007@hanmail.net

요 약

본 연구에서는 LVQ의 성능 개선을 위하여 SOM을 전처리로 이용하여 LVQ 네트워크를 설계한다. 제안한 설계 방법에선 SOM을 이용하여 LVQ 네트워크의 출력 뉴런 수와, reference vector의 초기값을 결정한다. SOM을 전처리로 이용한 LVQ를 Fisher의 Iris 데이터에 분류에 적용한 결과 기존의 LVQ보다 분류 성능이 뛰어났다.

ABSTRACT

We design LVQ network using SOM network for the LVQ's performance improvement. Reference vectors and the number of output neurons, they are the proposed LVQ network's initial parameters, are determined in SOM which is used for preprocessing LVQ. We simulate it to the grouping test of Fisher's Iris data. In this result, we confirm proposed LVQ network is better than existing LVQ in grouping test.

키워드

LVQ, SOM, class, subclass

Ⅰ. 서 론

SOM 네트워크는 패턴 공간상에서 서로 유사한 패턴끼리 묶어 줌으로써 패턴 공간을 분할한다.

입력으로 들어온 패턴과 출력 뉴런들과의 유클리드 방식에 의한 거리 측정에서 입력과 가장 가까운 출력 뉴런 한 개만이 승리 뉴런이 되며, 승리 뉴런으로부터 일정한 거리 내에 위치한 입력 패턴들을 하나의 클러스터로 분류하는 데 우수한 성능을 갖는다.[1][2][3]

패턴을 보다 정밀하게 분류하기 위하여 하나의 클러스터 혹은 클래스에 여러 개의 서브 클래스를 둘 수 있다. 따라서 학습을 통하여 서브 클래스들을 분류하고 분류된 서브 클래스들을 하나의 클래스로 지정한다면 패턴 분류 성능을 보다 정밀하고 정확하게 할 수 있다. 이러한 클래스 속의 서브 클래스를 학습시켜서 클래스로 분류하는 신경회로망이 바로 LVQ 네트워크이다.[4][5]

만일 동일한 클래스의 패턴 벡터들이 패턴 공간 상에서 여기 저기 흩어져서 서브 클래스를 형성하는 경우, SOM 네트워크는 패턴을 올바르게 분류할 수 없지만 LVQ 네트워크는 패턴 벡터들을 부분 부분의 서브 클래스로 학습을 시켜, 각 서브 클래스를 하나의 클래스로 묶음으로써 패턴을 올바르게 분류할 수 있다.[6][7]

그러나 이러한 LVQ 네트워크에도 설계할 적에 몇 가지 선행적으로 처리해야 할 문제점이 있다.

첫째로, LVQ 네트워크에서 서브 클래스 수를 얼마로 할 것이며, 각 출력 뉴런을 어떤 서브 클래스로 지정할 것인지 결정하는 것이다. 두번째 문제점은 네트워크의 reference vector의 초기값을 어떻게 정하는가

하는 것이다.

본 논문에서는 이러한 문제점을 해결하기 위해서, LVQ 네트워크의 전처리용 네트워크로써 SOM을 이용하는 방식을 제안한다. 제안한 방식은 입력 패턴 벡터를 먼저 SOM을 이용하여 분류를 한 후 학습에서 얻은 SOM의 출력 뉴런의 연결 강도를 LVQ의 초기 reference vector로 사용하며, 패턴 공간이 분할 되는 특성을 통해 LVQ 네트워크의 서브 클래스 수를 결정한다. 이렇게 전처리를 통해 얻은 결과로 LVQ 네트워크를 설계한다.

성능을 확인을 위하여 Fisher의 Iris 데이터 분류에 적용하였다.

II. SOM 과 LVQ

2-1. SOM

1984년 Kohonen이 제안한 SOM은 비지도 학습(unsupervised learning) 알고리즘의 일종으로 단일 경쟁 뉴런층으로 구성된 신경회로망이다. 경쟁 뉴런은 1차원이나 2차원 혹은 그 이상의 차원에서 물리적으로 정돈되어 있고, 각 뉴런은 특정 반경의 이웃 뉴런을 가지게 된다.[1][2]

일반적인 경쟁학습에서 뉴런들은 승리 뉴런을 결정하고, 승리 뉴런의 연결 강도를 강화하는 winner-takes-all 방식을 취한다. SOM이 일반적인 경쟁 학습과 다른 점은 승리 뉴런의 연결 강도 뿐 아니라 이웃 뉴런의 연결 강도까지 갱신한다는 점이다. 학습이 진행되는 동안 결과적으로 승리 뉴런과 이웃하는 뉴런들은 비슷한 연결 강도를 가지게 되고, 승리 뉴런과 이웃 뉴런들은 유사한 입력 벡터에 반응하게 된다.[1][2][3]

2-2. LVQ

LVQ의 기본 네트워크 구조는 SOM과 근본적으로 같지만, 다음과 같은 차이점이 있다. 출력 뉴런에 대하여 가정된 이웃반경이 없고, 각각의 출력 뉴런은 서브 클래스를 나타내고, 이 서브 클래스는 알려진 클래스에 속한다.[4]

학습을 하는 동한 LVQ의 출력 뉴런들은 이론적인 Bayes 분류기의 결정 표면에 접근하고, 학습 후에 LVQ는 reference vector와 입력 벡터가 가장 가깝게 위치하는 출력 뉴런으로 입력 벡터를 분류한다. 따라서 입력 벡터의 클래스와 winner neuron의 클래스가 같으면 연결 강도의 학습은 새로운 입력 벡터의 방향으로 향하고, 입력 벡터의 클래스와 winner neuron의 클래스가 다르면 연결 강도의 학습은 새로운 입력 벡터와 반대방향으로 이동한다.[5][6][7]

일반적으로 reference 벡터를 초기화 하는 방법에는 LVQ 네트워크의 출력 뉴런 수와 같은 수의 입력 벡터를 reference 벡터로 사용하고 나머지 입력 벡터를 학습에 사용하는 방법, 초기reference 벡터를 랜덤하게 지정하는 방법 등이 있다.

III. SOM을 이용한 LVQ 네트워크 설계

서론에서 기술한 바와 같이 LVQ 네트워크를 설계하기 위하여 클래스를 몇 개로 할 것이며, 각 클래스 내에 몇 개의 서브 클래스를 둘 것인가 하는 문제와 reference 벡터의 초기값을 어떻게 설정할 것인가를 결정하기 위하여 전처리 용도의 네트워크를 SOM을 이용하여 구현하였다.

SOM을 이용한 제안된 LVQ 전처리 네트워크 설계 절차는 크게 두 단계로 구성된다. 첫 번째 단계는 1차원 구조의 SOM을 설정하고, 입력 벡터로 학습시켜, 입력 벡터의 클러스터를 구한다. 중복된 승리 뉴런의 수가 없거나 적은 출력 뉴런의 수를 정한다. 이렇게 정한 출력 뉴런의 수가 LVQ의 서브 클래스의 수로 한다. 두 번째 단계는 SOM의 학습된 연결 강도를 LVQ 네트워크의 reference 벡터의 초기 값으로 정하고, 입력 벡터를 이용하여 LVQ를 학습시킨다. 제안한 알고리즘은 다음과 같다.

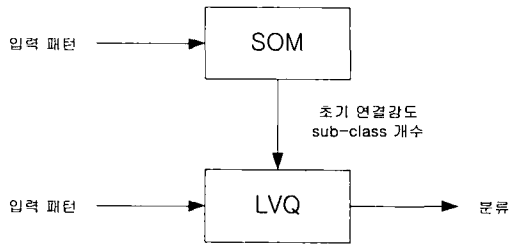

그림 . 전체 블록 선도

Step 1. $t = 0$

- SOM의 ${}^{SOM}w_i(0)$ 를 랜덤하게 선택

-많은 뉴런을 포함하도록 $N_C = N_C(t)$ 의 초기값 $N_C(0)$ 를 설정

- ${}^{SOM}\alpha(t)$ 의 초기값 설정

Step 3. SOM의 종료 조건이 만족되지 못하면

- do Step 4~8

Step 4.

- 각 입력 패턴 $\mathbf{X}$ 에 대하여

- do Step 5~8

Step 5.

- 유클리이드 거리를 계산

$$^{SOM}d(j)=\sum_i\left(^{SOM}w_{ji}-x_i\right)^2$$

Step 6.

- winner 뉴런 $^{SOM}y_j$를 찾는다.

Step 7.

- 고정 반경 내에서 연결 강도 조정

$$^{SOM}w_{ji}^{k+1}=^{SOM}w_{ji}^{k}+^{SOM}\alpha\left[x_i-^{SOM}w_{ji}^{k}\right]$$

Step 8.

- $N_C=N_C(t)$ 감소
- $^{SOM}\alpha(t)$ 감소

Step 9.

- SOM의 종료 조건 검사

Step 10.

-연결 강도 초기화 $^{LVQ}w_i(0)=^{SOM}w_i$

- $^{LVQ}\alpha(t)$의 초기값 설정
- 서브클래스 결정

Step 11. LVQ 종료 조건이 만족되지 못하면

- do Step 12~16

Step 12.

- 각 입력 패턴 **x**에 대하여
- do Step 13~15

Step 13.

- 유클리이드 거리를 계산

$$^{LVQ}d(j)=\sum_i\left(^{LVQ}w_{ji}-x_i\right)^2$$

Step 14.

- winner 뉴런 $^{LVQ}y_j$를 찾는다.

Step 15.

-입력 벡터의 클래스와 winner neuron의 클래스가 같으면

$$^{LVQ}w_{ji}^{k+1}=^{LVQ}w_{ji}^{k}+^{LVQ}\alpha\left[x_i-^{LVQ}w_{ji}^{k}\right]$$

-입력 벡터의 클래스와 winner neuron의 클래스가 다르면

$$^{LVQ}w_{ji}^{k+1}=^{LVQ}w_{ji}^{k}-^{LVQ}\alpha\left[x_i-^{LVQ}w_{ji}^{k}\right]$$

Step 16.

- 학습율 $^{LVQ}\alpha(t)$ 감소

Step 17.

- LVQ 종료 조건 검사

Ⅳ. 시뮬레이션

제안한 방식에 대해서 Fisher의 Iris 데이터를 분류하는 시뮬레이션을 수행하였다. Fisher의 Iris 데이터는 세 가지 품종 'Setosa', 'Versicolor', 'Virginica '의 붓꽃(iris)으로부터 각각 50개, 총 150개의 개체들을 추출해서 측정한 자료이다. 측정 변수는 꽃받침 조각의 길이(Sepal Length), 꽃받침 조각의 폭(Sepal Width), 꽃잎의 길이(Petal Length), 꽃잎의 폭(Petal Width)이다.

<Fisher's Iris Data Set>

Sepal-Length Sepal-Width Petal-Length Petal-Width / Species

Species: 1='Setosa' 2='Versicolor' 3='Virginica'

.

50 34 15 02 1

44 29 14 02 1

50 20 35 10 2

.

성능 확인을 위해 기존의 LVQ 네트워크와 비교 하였고, 각각의 네트워크를 학습시키기 위해 IRIS 데이터 75개를 사용했으며 나머지 75개는 테스트용으로 했다. 비교 대상은 데이터 에러 수로 했다.

4-1. 서브 클래스의 수에 따른 비교

기존 LVQ의 설계는 입력을 4개, 출력은 5에서 50으로 점차적으로 늘였으며 초기 연결강도는 [0,1]사이에서 렌덤하게 선택했다. 학습율은 0.05로 고정했으며 1000회 학습 시켰다.

제안한 LVQ의 SOM 구성은 입력을 4개, 출력은 5에서 50으로 점차적으로 늘였으며 초기 연결강도는 [0,1]사이에서 렌덤하게 선택했다. 학습율은 0.9에서 점차적으로 감소시켰으며 이웃반경 역시 최대 반경에서 1까지 점차적으로 감소시켰다. 학습은 10,000번 시켰다. 그리고 LVQ의 구성은 기존의 것과 동일하나 초기 연결강도는 SOM의 연결강도를 사용한다.

<학습 횟수 1000회 고정>

출력뉴런수	5	10	20	30	40	50
기존 LVQ 에러	8	6	7	7	26	29
제안한 LVQ 에러	4	4	2	2	6	8

4-2. 학습 횟수에 따른 비교

기존 LVQ의 설계는 입력을 4개, 출력은 30으로 고정했으며 초기 연결강도는 [0,1]사이에서 렌덤하게 선택했다. 학습율은 0.05로 고정했고 500에서 4000회까지 점차적으로 늘여가며 학습시켰다.

제안한 LVQ의 SOM 구성은 입력을 4개, 출력은 30으로 고정했으며 초기 연결강도는 [0,1]사이에서 렌덤하게 선택했다. 학습율은 0.9에서 점차적으로 감소시켰고 이웃반경 역시 최대 반경에서 1까지 점차적으로 감소시켰다. 학습은 10,000번 시켰다. 그리고 LVQ의 구성은 기존의 것과 동일하나 초기 연결강도는 SOM의 연결강도를 사용했다.

<출력 뉴런 30개> (단위: ×100)

iteration	5	10	15	20	30	40
기존 LVQ 에러	29	6	3	2	2	2
제안한 LVQ 에러	2	2	1	1	2	1

V. 결 론

본 논문에서는 LVQ의 성능 개선을 위하여 SOM을 전처리로 이용한 LVQ 네트워크 설계 방법을 제안하였다. 제안한 방법에서는 전처리 과정으로 SOM을 이용하여 LVQ의 초기 reference vector와 출력 뉴런 수를 결정하였다. IRIS 데이터를 대상으로 분류 테스트를 해 본 결과 기존의 LVQ보다 분류 성능이 향상됨을 알 수 있다.

참고문헌

[1] M. T. Hagan, H. B. Demuth, Mark Beale, Neural Network Design, PWS Publishing Company, 1995.

[2] V. E. Neagoe, A. D. Ropot, "Concurrent self-organizing maps for pattern classification ", First IEEE International Conference on Cognitive Informatics, pp. 304-312, 2002.

[3] N. R. Pal, J. C. Bezdek, E. C. K. Tsao, "Generalized clustering networks and Kohonen's self-organizing scheme", IEEE Transactions on Neural Networks, Vol. 4, pp. 549-557, 1993.

[4] A. Sato, k. Yamada, j. Tsukumo, "A multi-template learning method based on LVQ", IEEE International Conference on Neural Networks, Vol.2, pp. 632-637, 1993.

[5] Su-Jeong You, Chong-Ho Choi, "LVQ with a weighted objective function", IEEE International Conference on Neural Networks, Vol. 5, pp. 2763-2768, 1997.

[6] Xu Yong, Yan Guangqun, Chen Hexin, Dai Yisong, "A new competitive learning algorithm for vector quantization based on the neuron winning probability", IEEE International Conference on Intelligent Processing Systems, Vol. 1, pp. 485-488, 1997.

[7] Ching-Tang Hsieh, Mu-Chun Su, Uei-Jyh Chen, Horng-Jae Lee, "A new generalized learning vector quantization algorithm", IEEE Asia-Pacific Conference on Circuits and Systems, pp. 339-344, 2000.