

최근접 이웃망에 의한 참조벡터 학습

(Learning Reference Vectors by the Nearest Neighbor Network)

金伯變*

(Baek-Sep Kim)

要約

최근접 이웃 인식방법은 간단하고 인식오차가 근사적으로 Bayes오차의 2배가 넘지 않으므로 널리 이용된다. 그러나 이는 모든 학습패턴을 그대로 참조벡터로 사용하므로 학습패턴의 수가 늘어남에 따라 기억공간 및 인식시간이 늘어나는 문제가 있었다. 이에 참조벡터를 학습하는 LVQ(Learning Vector Quantization)가 제안되었으나 경험적 규칙을 사용하여 이론적 기반이 취약하고 반복횟수가 많으며 종료조건이 주어지지 않는 등의 문제가 있다.

본 논문에서는 오차함수를 정하고 이를 최소화하는 새로운 참조벡터 학습방법을 제시한다. 제안된 방법은 최근접 이웃 방법을 망 형태로 바꾸어 참조벡터를 노드사이의 연결강도로 표현하고 최대 경사법을 이용하여 연결강도를 학습한다. 그 결과 제안된 방법은 한번 반복 때마다 학습되는 참조벡터의 수가 LVQ보다 많아 반복횟수가 줄어들며, 인식오차가 LVQ보다 작아짐을 보인다.

Abstract

The nearest neighbor classification rule is widely used because it is not only simple but the error rate is asymptotically less than twice Bayes theoretical minimum error. But the method basically use the whole training patterns as the reference vectors, so that both storage and classification time increase as the number of training patterns increases. LVQ(Learning Vector Quantization) resolved this problem by training the reference vectors instead of just storing the whole training patterns. But it is a heuristic algorithm which has no theoretic background, there is no terminating condition, and it requires a lot of iterations to get the meaningful result.

This paper is to propose a new training method of the reference vectors, which minimize the given error function. The nearest neighbor network, the network version of the nearest neighbor classification rule, is proposed. The network is functionally identical to the nearest neighbor classification rule, and the reference vectors are represented by the weights between the nodes. The network is trained to minimize the error function with respect to the weights by the steepest descent method. The learning algorithm is derived and it is shown that the proposed method can adjust more reference vectors than LVQ in each iteration. Experiment showed that the proposed method requires less iterations and the error rate is smaller than that of LVQ2.

*正會員, 翰林大學校 電子計算科
(Dept. of Comp. Science, Hallym Univ.)

接受日字 : 1993年 9月 20日

1. 서론

패턴인식에서 최근접 이웃 방법(Nearest Neighbor Method)은 참조벡터(reference vector)들을 저장한 후, 인식하고자 하는 패턴(시험 표본)이 들어 오면 그에 가장 가까운 참조벡터를 계산하여 그 참조벡터가 속한 집단에 시험표본이 속한다고 판단하는 것이다.^[1] 이 방법은 개념적으로 간단하고 각 집단의 확률분포함수에 무관하게 근사적으로 Bayes오차의 2배가 넘지 않는다는 사실이 증명되어 패턴인식 분야에서 널리 사용되고 있다.^[2] 그러나 최근접 이웃 방법은 모든 학습패턴들이 참조벡터로 사용되므로 학습패턴의 수가 늘어나면 기억공간과 계산량이 비례적으로 늘어나는 문제가 있다. 이를 부분적으로 극복하기 위해 학습패턴중 일부만을 참조벡터로 사용하는 방법들^[3,4]이 연구되었으며 최근접 참조벡터를 빨리 찾는 알고리즘들^[5,6]이 제안되었으나 근본적인 문제를 해결하지는 못했다.

Kohonen^[7]은 참조벡터를 구하는 새로운 방법으로 LVQ(Learning Vector Quantization)을 제안하였다. 이는 학습패턴 전부, 혹은 일부를 참조벡터로 사용하는 대신에 학습규칙을 통해 참조벡터를 결정하는 것이다. LVQ는 참조벡터의 학습이라는 새로운 개념을 제공하였으며 이렇게 함으로써 참조벡터의 갯수는 학습패턴의 수와 무관하게 사용자가 결정할 수 있게 되었다. LVQ는 현재까지 알려진 유일한 참조벡터 학습 방법으로^[13] 여러가지 변형이 있는데 이중 통계적 패턴인식에는 LVQ2가 가장 적합하다고 알려져 있다.^[8]

LVQ는 다른 인식 방법에 비해 다음과 같은 장점이 있어 음성인식^[9], 문자인식^[10], Radar신호 분별^[11], 등 여러 분야에서 이용되고 있다. 첫째, LVQ의 참조벡터들은 오류역전과 신경회로망^[12]의 은닉노드들과 흡사한 역할을 하는데 후자는 집단이름(class label)이 없는데 비해 LVQ의 참조벡터는 그것이 속한 집단이름을 가지고 있다. 이는 여러가지 인식방법을 혼합 사용하여 인식 시스템을 구성하는 경우 큰 장점이 된다. 둘째 기존의 최근접 이웃 방법에 비해 참조벡터 수를 매우 작게 할 수 있어 필요한 기억공간이 작으며 인식시간이 적게 걸린다는 장점이 있다. 마지막으로 오류 역전파 방법에 비해 학습 규칙이 간단할 뿐 아니라 인식과정도 간단하고 인식시간이 적게 걸린다. 더구나 최근접 이웃을 구하는 빠른 알고리즘^[5,6]이 개발되어 있으므로 이를 이용하면 인식시간을 더욱 빠르게 할 수 있다.

그러나 LVQ는 기본적으로 경험적 규칙(heuristic

rule)으로써 오류 역전파에서와 같이 목적함수를 최소화하는 등의 이론적 체계가 주어지지 않다. 또한 반복적 알고리즘인데 종료조건이 주어지지 않아 반복횟수를 결정하기 힘들고, 한번 반복때마다 최대 2개의 참조벡터만이 학습되므로 반복횟수가 많다는 단점이 있다.

본 논문은 이들 단점을 극복하기 위해 참조벡터를 학습하는 체계적인 방법을 제안하고 이를 LVQ와 비교하고자 하는 것이다. 이를 위하여 최근접이웃 방법을 망 형태로 재구성한 최근접 이웃망을 제안하고 이 망에서 참조벡터를 학습하는 학습규칙을 제시한다.

II. 최근접 이웃 방법의 망 형태 표현

기존의 최근접 이웃 방법은 시험표본(test sample)을 인식하기 위해 다음의 과정을 거친다.

- 1) 입력과 각 참조벡터와의 거리를 계산한다.
- 2) 거리가 가장 작은 참조벡터를 구한다.
- 3) 그 참조벡터가 속하는 집단이름을 출력한다.

이 과정을 전향적 망(feed-forward network) 형태로 표시한 것을 최근접 이웃 망(Nearest Neighbor Network)라고 부르기로 하자. 이의 구조를 그림 1에 보인다.

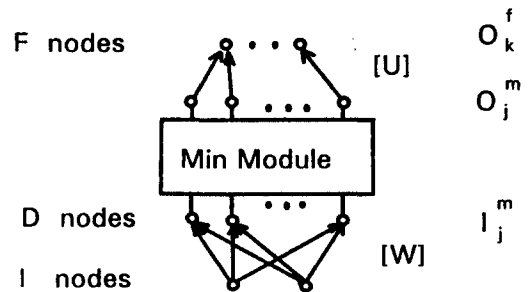


그림 1. 최근접 이웃망의 구조
Fig. 1. Structure of the Nearest Neighbor Network.

그림에서 I 노드들은 입력을 받아들이는 노드들로서 p개가 존재하며 출력값은 입력값과 같다. 이후 i 번째 I노드의 출력값을 I_i 로 표현하기로 한다. D노드는 입력과 참조벡터사이의 거리를 계산하는 노드들로서 j번째 D노드의 출력은 Min module의 j번째 입력이 되며 이를 I_j^m 으로 표현한다. 즉

$$I_j^m = \sum_{u=1}^p (I_u - W_{uj})^2 \tag{2-1}$$

이다. 여기서 w_{ij} 는 i 번째 I노드와 j 번째 D노드사이의 연결강도이다. j 번째 D노드는 각각의 입력노드들과 $w_{1j}, w_{2j}, \dots, w_{mj}$ 의 연결강도들로 연결되어 있는데 이들은 w_j 를 구성하며 j 번째 참조벡터라는 의미를 갖는다. D노드의 갯수 q 는 사용자가 정한 참조벡터의 갯수가 되며 D노드들의 출력, 즉 Min module의 입력들은 각 참조벡터와 입력사이의 거리의 제곱이 된다. 입력과 D노드들 사이의 연결강도들은 q 개의 참조벡터로 이루어진 매트릭스 $[W]$ 로 표현할 수 있는데 이들은 3장에서 보이는 학습방법에 의해 학습된다.

Min module은 입력값이 최소가 되는 출력 노드를 알아내는 신경망이다. 이는 최대값대신 최소값을 선택한다는 점을 제외하고는 Hamming Net 혹은 기타 경쟁학습을 사용하는 신경망에 사용되는 MAXNET^[1] 과 동일하다. MAXNET은 이웃 노드들에게 억제성 자극을 전달하여 재귀적으로 계산하는 방법^[2], 크기를 비교하는 신경망을 계층적으로 배열한 다층 전향망을 사용하는 방법^[15], AND 연산을 하는 노드들로 단층 전향망을 구성하는 방법^[16] 등이 있다. 이들 방법들은 출력이 입력에 대해 미분 불가능하므로 본 논문에서는 미분 가능한 MINNET(모든 MAXNET은 입력의 부호를 바꾸거나 연결강도의 부호를 바꾸어 MINNET으로 사용할 수 있다)을 제안하고 이를 Min module이라고 한다.

Min module은 q 개의 입력과 q 개의 출력으로 구성된다. Min module에 q 개의 입력들이 주어지면 j 번째 노드의 출력

$$O_j^m = \frac{\prod_{u=1}^q \frac{1}{u+j} + e^{-\frac{(I_u^m - I_u^m)\sigma}{u+j}}}{\sum_{i=1}^q \prod_{u=1}^q \frac{1}{i+u} + e^{-\frac{(I_u^m - I_u^m)\sigma}{i+u}}} \quad (2-2)$$

로 정의한다. 여기서 분자는 sigmoid함수를 곱한 형태로서 j 번째 입력을 다른 입력들과 비교하는 역할을 하며 분모는 모든 출력의 합이 1이 되도록 정규화한 것이다. σ 는 상수로 sigmoid함수의 활성화 영역의 폭을 결정하는 역할을 한다. $\sigma = 0$ 인 경우 분자의 하나의 항은 $I_u^m \leq I_u^m$ 이면 1이고 그렇지 않으면 0이 된다. 이러한 항들을 곱하는 것은 AND연산과 같은 효과를 나타내므로 모든 $u \neq j$ 에 대해 곱하면 입력이 가장 작은 노드만 출력이 1이되고 나머지는 모두 0이 된다.

Min module은 다음과 같은 특성을 갖는다.

- 1) 출력의 범위는 $[0, 1]$ 이며 해당 입력(j 번째 출력에 대해서는 I_j^m)에 대해 연속적이고 미분 가능하다.

- 2) 출력값들은 해당되는 입력값의 역순이 된다. 즉 모든 i, j 에 대해 $I_i^m < I_j^m$ 이면 $O_i^m > O_j^m$ 이다.
- 3) $\sigma = 0$ 이 되면 q 개의 입력중 가장 작은 값을 갖는 노드(이를 임의로 k 번째 노드라고 하자)에 해당되는 출력노드의 값(O_k^m)만이 1이 되고 나머지는 모두 0이 된다. 즉 입력값이 최소인 노드를 찾는다.

앞에서 Min module의 입력값은 각각 I노드에 들어오는 입력과 $[W]$ 에 기억된 참조벡터들과의 거리의 제곱이라고 하였으므로 Min module은 거리가 가까운 순으로 참조벡터를 정렬하는 역할을 한다.

Min module의 출력은 F노드들과 $[U]$ 로써 연결된다. $[U]$ 는 참조벡터가 속한 집단이름을 인가하는 역할을 한다. 즉 참조벡터 w_k 가 k 번째 집단에 속해 있다면(2-3)

$$u_j^i = \begin{cases} 1, & \text{if } I = k \\ 0, & \text{otherwise} \end{cases} \quad (2-3)$$

이다. $[U]$ 는 초기 참조벡터를 정할 때 결정되며 이후 학습에 의해 변하지 않는다. F노드 각각은 하나의 집단을 나타내는 것으로서 인식하려는 집단의 개수(임의로 r 개 라고 하자)만큼 존재한다. k 번째 F노드의 출력을

$$O_k^f = \sum_{j=1}^q u_{jk} O_j^m \quad (2-4)$$

으로 정의한다.

이상에서 $\sigma = 0$ 이고 참조벡터들을 각각 학습패턴의 좌표값으로 정한 경우 최근접 이웃망의 인식과정은 다음과 같다.

- 1) D노드 각각은 시험표본 입력과 참조벡터 사이의 거리를 계산한다.
- 2) Min module은 이들중 거리가 가장 작은 노드에만 출력값을 1을 내주고 나머지 노드들의 출력값은 0이 되게한다.
- 3) F노드중 출력값이 1인 Min module 노드와 연결된 노드만 1값을 가지고 나머지노드들은 0이 된다.

이 과정은 앞에서 설명한 최근접 이웃 인식방법과 동일하므로 같은 인식 결과를 내주게 된다. LVQ도 최근접 이웃 인식방법을 사용하고 있으므로 인식결과 는 $\sigma = 0$ 인 최근접 이웃망과 같다.

LVQ와 최근접 이웃망은 모두 인식과정에서 최근접 방법을 사용한다는 점은 동일하나 참조벡터의 학습과정에서 차이를 보인다. LVQ는 초기참조벡터를 사용

자가 임의로 설정한 후 학습패턴으로 부터 경험적 학습규칙을 통해 반복적으로 바꾸어 나가는 것으로서 LVQ1, LVQ2, LVQ3, OLVQ등이 제안되어 있다.^[14] 이중 통계적 패턴인식에 가장 적합하다고 알려져 있는^[8] LVQ2의 학습과정은 다음과 같다.

- 1) 학습패턴중 하나를 고른다. 이를 x 라고 하자.
- 2) 참조벡터중 x 에 가까운 순서로 두개를 선택한다. 이를 각각 w_i, w_j 라 하고 와이들의 거리를 각각 d_i, d_j 그리고 이들이 속한 집단을 각각 c_i, c_j 라고 하자.
($d_i \leq d_j$ 이다.)
- 3) x 가 window안에 있고 $x \in c_i$ 이며 $c_j \neq c_i$ 라면

$$w_i(t+1) = w_i(t) + \alpha(t)[x(t) - w_i(t)] \quad (2-5)$$

$$w_j(t+1) = w_j(t) - \alpha(t)[x(t) - w_j(t)]. \quad (2-6)$$

로 바꾼다.

- 4) 1)부터 3)까지 주어진 횟수만큼 반복한다.
- 위에서 $\min(d_i/d_j, d_i/d_i) > s$ 이면 s 는 window안에 있다고 정의한다. 여기서 s 는 window폭을 결정하는 상수이다. α 는 0과 1사이의 값을 가지며 시간에 따라 단조감소한다.

최근접 이웃망에 대한 학습은 LVQ2에 비해 다소 복잡하다. 앞에서 설명한 바와 같이 최근접 이웃망은 다층 퍼셉트론과 같이 전향적 망 구조를 가지며 각 노드 혹은 module는 미분 가능하다. 그러므로 다층 퍼셉트론에서의 학습규칙인 오류역전파(Error back propagation)방법과 같은 학습규칙을 사용할 수 있다. 즉, 출력에서 오차를 계산하여 이를 최소화 하는 방향으로 연결가중치(참조벡터)를 변경하는 것이다. 자세한 과정을 다음 장에 보인다.

Ⅲ. 최근접 이웃망의 학습

학습패턴은 입력값과 그것이 속한 집단이름으로 구성되는데 집단이름은 학습을 위한 교사입력(teaching input)역활을 한다. 학습패턴으로부터 참조벡터들을 학습하는 과정은 다음과 같다. 먼저 입력값을 최근접 이웃망의 I노드들에 인가하여 F노드의 출력 O_k' 들을 구한다. O_k' 를 교사입력 t_k 과 비교하면 다음과 같은 자승오차를 구할 수 있다.

$$E = \sum_{k=1}^r (O_k' - t_k)^2 / 2 \quad (3-1)$$

여기서 상수 2는 계산상 편의를 위하여 임의로 넣은 것이다. E를 최소화하는 w_{ij} 를 구하기 위해 오류 역

전과 학습 알고리즘^[12]에서와 같이 최대 경사법(Steepest Descent Method)를 사용하면 w_{ij} 는 한번 반복때마다 Δw_{ij} 를 더한 값으로 변경된다. 여기서

$$\Delta w_{ij} \propto - \frac{\partial E}{\partial w_{ij}} \quad (3-2)$$

이다. $\frac{\partial E}{\partial w_{ij}}$ 를 구하는 과정을 아래에 보인다. 앞에서 설명한 바와같이 I_i, I_j^m, O_j^m, O_k' 는 각각 i 번째 I노드의 출력값, j 번째 D노드의 출력값(즉, Min module의 j 번째 입력값), Min module의 j 번째 출력값, 그리고 k 번째 O노드의 출력값을 나타냄에 유의하기 바란다.

미분의 연결규칙(chain rule)에 의해

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial I_j^m}{\partial w_{ij}} \frac{\partial E}{\partial I_j^m} \quad (3-3)$$

이되고 우변의 첫항은 (2-1)으로부터

$$\frac{\partial I_j^m}{\partial w_{ij}} = w_{ij} - I_i \quad (3-4)$$

가 된다. (상수 2는 무시한다) 둘째항은 다시

$$\frac{\partial E}{\partial I_j^m} = \sum_{u=1}^q \frac{\partial O_u^m}{\partial I_j^m} \frac{\partial E}{\partial O_u^m} \quad (3-5)$$

로 쓸 수 있는데 여기서

$$\frac{\partial O_u^m}{\partial I_j^m} = \Delta(j, u) \quad (3-6)$$

라고 정의하자. $\Delta(i, j)$ 는 다음과 같이 구해진다. 자세한 계산과정은 부록을 참조하기 바란다.

$$A(j) = \prod_{u \neq j} \frac{1}{1 + e^{-(I_u^m - I_j^m)/\sigma}}, \quad (3-7)$$

$$B = \sum_{u=1}^q A(u), \quad (3-8)$$

그리고

$$C(u, v) = \frac{e^{-(I_u^m - I_v^m)/\sigma}}{1 + e^{-(I_u^m - I_v^m)/\sigma}} \quad (3-9)$$

라고 하면 $\Delta(i, j)$ 는

$$- \frac{O_i^m}{\sigma} \frac{1}{B} \left[\sum_{u=1}^q A(u) \sum_{u=1}^q C(u, 1) + \sum_{u=1}^q A(u) C(u, 1) \right], \text{ for } i = j \quad (3-10)$$

$$\frac{O_j^m}{\sigma} \Delta(i, i) + \frac{O_i^m}{\sigma} \left[A(j) C(i, j) + \sum_{u=1}^q C(u, i) \right], \text{ for } i \neq j$$

이다. (3-5)식의 마지막 항은 다시

$$\frac{\partial E}{\partial O_v^m} = \sum_{v=1}^q \frac{\partial O_v^f}{\partial O_v^m} \frac{\partial E}{\partial O_v^f} \quad (3-11)$$

로 쓸 수있다. 이 중 우변 첫항은 (2-4)식으로부터

$$\frac{\partial O_v^f}{\partial O_v^m} = u_{mv} \quad (3-12)$$

가 되며 둘째항은 (3-1)식으로부터

$$\frac{\partial E}{\partial O_v^f} = O_v^f - t_v \quad (3-13)$$

가 된다. 이상을 종합하면

$$\frac{\partial E}{\partial w_{ij}} = (w_{ij} - I_i) \sum_{u=1}^q \Delta(j,u) \sum_{v=1}^r u_{uv} (O_v^f - t_v) \quad (3-14)$$

이다. 이는 (3-2)식에서 보인 바와같이 참조벡터를 학습하는 양을 결정하게 되는데 (2-5)식 및 (2-6)식에서 처럼 학습입력과 참조벡터와의 차이에 비례한다는 점은 LVQ2와 같으나 다음과 같은 차이가 있다. 첫째로는, (3-14)식에서 $O_v^f - t_v$ 는 망의 출력값과 교사입력과의 차이 즉 오차를 나타내는데 오차가 없으면 참조벡터를 바꾸지 않는다. 다른 하나는 LVQ2에서는 하나의 학습입력에 대해 가까운 두개의 참조벡터들만 학습되었으나 제안된 방법은 모든 참조벡터들(실제로는 학습입력과 거리가 매우 먼 참조벡터들의 학습량은 무시할 정도로 작게 되며 이들의 숫자는 가변적이다)이 학습된다는 점이다.

윗식의 계산시간은 주로 (3-10)식의 계산이 차지한다. (3-10)식을 살펴보면 $O_v^m=0$ 이면 $\Delta(i,j)=0$ 임을 알 수 있는데 이는 Min module의 출력이 0인 노드에 대해서는 σ 를 계산할 필요가 없다는 것을 의미한다. Min module의 출력은 2장에서 언급한 바와 같이 학습입력과 참조벡터들 사이의 거리를 구하여 거리가 가까운 순으로 정렬한 값이며 σ 가 매우 크지 않으면 몇개를 제외하고는 이 값은 대부분 거의 0이 된다. 특히 $\sigma=0$ 인 경우는 단 한개만이 1의 값을 가지고 나머지는 모두 0이 된다. 그러므로 계산시간을 줄이기 위해 Min module의 출력값에 임계치(실험에서는 10^{-6} 로 하였다)를 정하여 이보다 값이 작은 경우 (3-10)식은 계산하지 않는 것이 바람직하다. 또한 (3-9)식에서 $C(v,u) = 1 - C(u,v)$ 을 쉽게 유도할 수 있으므로 이를 이용하여 $C(u,v)$ 의 계산량을 반으로 줄일 수 있다. (3-10)식에서 분모에 있는 상수 σ 는 제거하였음을 아울러 알려준다. 이는 상수이므로 다음에 설명하는 학습률에 흡수될 수 있기 때문이다.

이상을 기초로 N개의 학습패턴이 주어진 경우 참

조벡터의 학습 알고리즘은 다음과 같다.

- 0) [W], [U] 를 초기화 한다. 이는 LVQ에서 초기 참조벡터를 결정하는 것과 같다. 또한 Err=0으로 놓는다.
- 1) 학습패턴중 하나를 인가하여 각노드의 값을 구한다. A()와 B는 Min module의 출력값을 계산하는 과정에서 얻어진다.
- 2) Min module의 출력값 O_v^m 이 임계치보다 큰 모든 j 에 대해 $\Delta(i,j)$ 를 계산한다. 여기서 i 와 j 는 1, 2, ..., q이다.
- 3) $w_{ij} = w_{ij} + \alpha(I_i - w_{ij}) \sum_{u=1}^q \Delta(j,u) \sum_{v=1}^r u_{uv} (O_v^f - t_v)$ (3-15)로 바꾼다. α 를 학습률(learning rate)라고 부르게 한다.
- 4) (3-1)식에서 E를 구하여 $Err = Err + 2E$ 로 한다.
- 5) 인가할 학습패턴이 남아있으면 1)로 간다.
- 6) N개의 학습패턴에 대한 평균 자승오차 $MSE = Err / N$ 이 원하는 값보다 작거나 반복횟수가 주어진 횟수를 넘으면 끝나고 아니면 $Err=0$ 으로 하고 1)로 간다.

IV. 실험 결과

실험을 위하여 Hart [3]에 의해 제안된 복잡한 결정 영역(decision boundary)문제를 택하였다. 이는 2차원 2집단 문제이나 결정영역이 복잡하여 최근접 이웃 인식문제의 성능을 비교하기 위해 자주 쓰인다. 각 집단의 학습패턴을 750개씩으로 한 경우 학습패턴과 결정영역을 그림 2에 보인다.

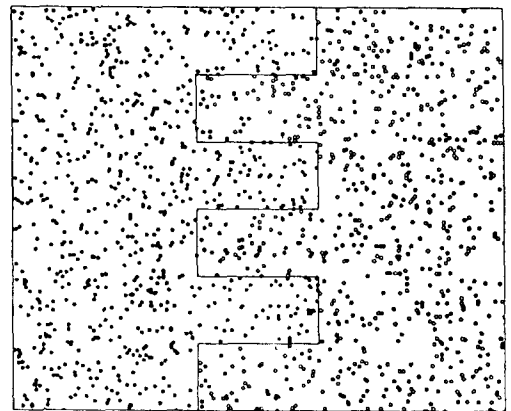


그림 2. 결정영역과 학습패턴
Fig. 2. Decision boundary and training patterns.

이 문제에서 K-means방법으로 초기 참조벡터를 정한 후^[9] LVQ2및 제안된 방법으로 학습한 다음의 인식오차를 그림 3에 보인다. 인식오차는 학습패턴과 같은 분포를 가진 시험표본들을 15,000개 생성하여 측정한 것이다. LVQ2의 경우는^[14]와 같이 초기 학습률을 0.05로 하여 반복 때마다 직선적으로 감소하도록 하였고 $s=0.3$ 으로 하였으며 반복횟수는 10,000으로 하였다. 제안된 방법의 경우는 학습률은 0.9로 고정하고 σ 는 초기치를 0.01로 하여 반복 때마다 1.5배씩 감소하도록 하였으며 반복회수는 8,250번으로 하였다. 또한 (3-15)식에서 맨 끝에 있는 $O_v^f - t_v$ 는 최근접 이웃망의 출력과 교사입력과의 오차를 나타내는데 이의 절대값이 0.1보다 작은 경우 0으로 대치하였다. 이는 오류역전파의 학습에서 학습속도를 빠르게 하기위해 자주 쓰이는 방법을 따른 것이다.^[12] 그림에서 제안된 방법이 전반적으로 인식오차가 적음을 알 수 있다. 그러나 학습 시간은 제안된 방법으로 8,250번 반복한 것이 LVQ2로 10,000번 반복한 것에 비해 약 4배 내지 8배 정도 더 걸렸다.

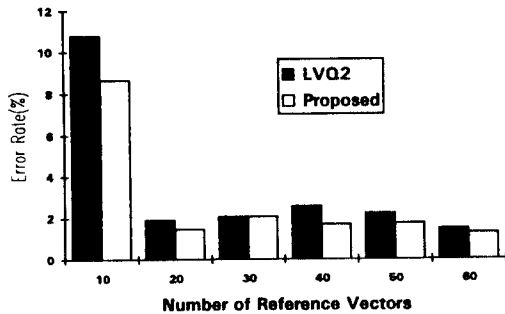


그림 3. 참조벡터 수에 따른 인식오차의 변화
Fig. 3. Change of recognition error due to number of reference vectors.

참조벡터의 수가 50개이고 종료조건을 MSE 0.01로 한 경우 최근접 이웃망의 학습이 진행되는 과정을 그림 4에 보인다. 여기서 횡축은 epoch수를 나타내는데 전체 학습패턴이 하나의 epoch를 형성하므로 한 epoch는 750번의 학습에 해당된다. 그림 4-a)와 b)는 각각 반복횟수에 따른 MSE의 변화와 인식오차의 변화를 나타낸다. MSE는 학습할 때 사용하는 오차이고 인식오차는 학습된 참조벡터를 사용하여 인식하였을 때의 실제 오차이다. 이들은 근본적으로 오차라는 점은 같으나 MSE는 학습패턴을 최근접 이웃망에 인가한 후 출력값과 교사입력과의 오차를 평균한 것이고 인식오차는 시험패턴을 최근접 이웃 방법으로

판별(0/1 decision)했을 경우의 오차라는 점에서 차이가 있다. 이들은 동일한 참조벡터를 사용하므로 변화 추세가 비슷할 것으로 기대되었으나 그림에서 MSE와 인식오차의 변화 추세는 서로 같지 않음을 알 수 있다. 이 차이는 학습패턴과 시험패턴은 같은 분포에서 뽑아서 냈고 표본의 수가 비교적 많으므로 표본 분포의 차이가 크지 않다고 가정하면 오차 측정 방법의 차이에서 주로 기인한다고 볼 수 있다. 본 논문에서는 LVQ와 같이 최근접 이웃망으로 학습된 참조벡터들을 사용하여 최근접 이웃 방법으로 인식하였으나 학습된 최근접 이웃망을 인식기로 사용하면 MSE와 인식오차는 비슷한 변화 추세를 나타내리라 생각된다.

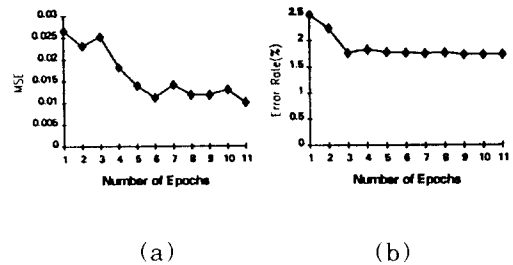


그림 4. 반복횟수에 따른 MSE의 변화(a)와 인식 오차의 변화(b)
Fig. 4. Changes of MSE(a) and Classification Error(b) as varying number of epochs.

V. 결론

LVQ는 최근접 이웃 패턴인식 방법에서 참조벡터를 학습하는 방법을 제안하여 기존의 최근접 이웃방법에서 학습패턴이 늘어남에 따라 기억공간과 인식시간이 늘어나는 문제를 해결하였다. 그러나 이는 경험적 학습규칙을 사용하여 이론적 기반이 취약하고 한번 반복때마다 최대 두개의 참조벡터만이 학습되어 반복횟수가 많고 종료조건이 주어지지 않다는 등의 문제가 있다.

본 논문에서는 참조벡터를 학습하는 새로운 방법을 제시하였다. 이를 위하여 미분가능한 MINNET인 Min module을 고안하고 Min module을 사용하여 기존의 최근접 이웃 방법을 망 형태로 표현한 최근접 이웃망을 제안하였다. 최근접 이웃망에서는 참조벡터들이 노드들 사이의 연결강도로 표현된다. 이들을 학습하기 위해 자승오차를 목적함수로 정했으며 최대 경사법을 이용하여 목적함수를 최소화하는 학습규칙

을 유도하였다. 이렇게 유도된 학습규칙은 이론적 체계가 명확하며 종료조건을 가지고 있을 뿐아니라 한번 반복될 때마다 학습되는 참조벡터의 수가 가변적으로 정해지므로 LVQ에 비해 반복횟수가 줄어든다는 장점이 있다.

복잡한 결정영역을 가지는 문제에 대해 실험한 결과 단지 8,250번(11 epochs)의 반복으로도 LVQ로 10,000번 반복한 경우보다 높은 인식률을 보여 반복횟수는 줄었으나 한번 반복하는데 시간이 오래 걸려 전체적으로는 4-8배의 학습시간이 소요되었다.

부 록

$\Delta(i, j)$ 즉, $\frac{\partial O_j^m}{\partial I_i^m}$ 을 계산하는 과정은 다음과 같다. $A(j)$, B , $C(u, v)$ 에 대한 정의는 각각 (3-7), (3-8) 그리고 (3-9)식을 참조하기 바란다. (2-2)식은 (3-7)식과 (3-8)식으로 부터 어떤 j 에 대해서도

$$O_j^m = \frac{A(j)}{B} \tag{A-1}$$

라고 쓸 수 있다. 그러므로

$$A'(j) = \frac{\partial A(j)}{\partial I_i^m} \tag{A-2}$$

$$B' = \frac{\partial B}{\partial I_i^m} \tag{A-3}$$

라면

$$\Delta(i, j) = \frac{A'(j)B - B'A(j)}{B^2} \tag{A-4}$$

이다. (A-4)식을 i 와 j 가 같은 경우와 그렇지 않은 경우로 나누어 계산하기로 하자.

1) $i = j$ 인 경우

$A(j)$ 에는 i 를 포함하는 항이 $m-1$ 개 있으므로 (3-7)식의 각 항을 I_i^m 으로 미분하여 모두 더하면

$$A'(i) = -\frac{1}{\sigma} A(i) \sum_{u \neq i}^q C(u, i) \tag{A-5}$$

가 된다. ($i=j$ 이므로 $A'(j)$ 대신 $A'(i)$ 로 적었다.) B 는 $A(i)$ 와 i 를 제외한 모든 u 에 대해 $A(u)$ 들의 합으로 쓸 수 있고 각 $A(u)$ 들은 (3-7)식에서 $m-1$ 개 항을 곱한 형태이나 이중 한 항만 I_i^m 를 포함하고 있으므로 $A(u)$ 를 I_i^m 으로 미분하면 $A(u)C(i, u)/\sigma$ 가 된다. 그러므로

$$B' = A'(i) + \frac{1}{\sigma} \sum_{u \neq i}^q A(u)C(i, u) \tag{A-6}$$

이다. (A-5)식과 (A-6)식으로 부터

$$\begin{aligned} A'(i)B - B'A(i) &= A'(i) \sum_{u \neq i}^q A(u) - A'(i)A(i) - \frac{A(i)}{\sigma} \sum_{u \neq i}^q A(u)C(i, u) \\ &= A'(i) \sum_{u \neq i}^q A(u) - \frac{1}{\sigma} \sum_{u \neq i}^q A(u)C(i, u) \\ &= -\frac{A(i)}{\sigma} \left[\sum_{u \neq i}^q A(u) \sum_{u \neq i}^q C(u, i) + \sum_{u \neq i}^q A(u)C(i, u) \right] \end{aligned} \tag{A-7}$$

이 된다. 그러므로 (A-1)식과 (A-7)식으로부터

$$\Delta(i, i) = -\frac{O_i^m}{\sigma} \frac{1}{B} \left[\sum_{u \neq i}^q A(u) \sum_{u \neq i}^q C(u, i) + \sum_{u \neq i}^q A(u)C(i, u) \right] \tag{A-8}$$

이 얻어진다.

2) $i \neq j$ 인 경우

j 가 i 와 같지 않으면 $A(j)$ 는 $m-1$ 개 항중 한 항만 I_i^m 를 포함하므로

$$A'(j) = \frac{1}{\sigma} A(j)C(i, j) \tag{A-9}$$

이 된다. (A-9)식과 (3-8)식으로부터

$$\begin{aligned} B' &= A'(i) + \sum_{u \neq i}^q A'(u) \\ &= -\frac{1}{\sigma} A(i) \sum_{u \neq i}^q C(u, i) + \frac{1}{\sigma} \sum_{u \neq i}^q A(u)C(i, u) \end{aligned} \tag{A-10}$$

이다. (A-9)식과 (A-10)식은

$$\begin{aligned} A'(j)B - B'A(j) &= \frac{1}{\sigma} [BA(j) + C(i, j) + A(i)A(j) \\ &\quad - \sum_{u \neq i}^q C(u, i) - A(j) \sum_{u \neq i}^q A(u)C(i, u)] \end{aligned} \tag{A-11}$$

을 내주므로 이것과 (A-1), (A-4)식을 이용하면

$$\Delta(i, j) = \frac{O_j^m}{\sigma} [A(j)C(i, j) + \frac{1}{B} A(i) \sum_{u \neq i}^q A(u)C(i, u)] \tag{A-12}$$

이 된다. 이것을 간단히 하기 위해 (A-7)식을 다시 정리한

$$\sum_{u \neq i}^q A(u)C(i, u) = -\frac{\sigma}{O_i^m} B \Delta(i, i) - \sum_{u \neq i}^q A(u) \sum_{u \neq i}^q C(u, i) \tag{A-13}$$

을 이용하자. (A-13)식을 (A-12)식에 대입하면

$$\begin{aligned} \Delta(i, j) &= \frac{O_j^m}{\sigma} [A(j)C(i, j) + \frac{1}{B} A(i) \sum_{u \neq i}^q C(u, i) \\ &\quad + \frac{1}{B} \sum_{u \neq i}^q A(u) \sum_{u \neq i}^q C(u, i)] + \frac{O_j^m}{O_i^m} \Delta(i, i) \\ &= \frac{O_j^m}{O_i^m} \Delta(i, i) + \frac{O_j^m}{\sigma} [A(j)C(i, j) + \sum_{u \neq i}^q C(u, i)] \end{aligned} \tag{A-14}$$

이 얻어진다. 윗 식의 마지막 과정은 (3-8)식에서

$$A(i) + \sum_{u \neq i}^q A(u) = \sum_{u \neq i}^q A(u) = B \quad (\text{A-15})$$

인 성질을 이용한 것이다.

參考文獻

- [1] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," IEEE Trans. Inform. Theory, vol. IT-13, pp. 21-27, Jan. 1967.
- [2] *NN Pattern Classification Techniques*, edited by B. V. Dasarathy, IEEE Computer Science Press, 1991.
- [3] P. E. Hart, "The condensed nearest neighbor rule," IEEE Trans. Inform. Theory, vol. IT-14, pp. 515-516, May 1968.
- [4] P. A. Devijver and J. Kittler, "On the edited nearest neighbor rule," Proc. 5th Int'l Conf. Pattern Recognition, pp. 72-78, Dec. 1980.
- [5] B. S. Kim and S. B. Park, "A fast k nearest neighbor finding algorithm based on the ordered partition," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. PAMI-8, no. 6, pp. 761-766, Nov. 1986.
- [6] A. J. Broder, "Strategies for efficient incremental nearest neighbor search," Pattern Recognition, vol. 23, no. 1, pp. 171-178, 1990.
- [7] T. Kohonen, "The self-organizing map," Proceedings of the IEEE, vol. 78, no. 9, pp. 1464-1480, Sept. 1990.
- [8] T. Kohonen, G. Barna and R. Chrisley, "Statistical pattern recognition with neural networks: Benchmarking studies," IEEE Int'l Conf. Neural Networks, vol. I, pp. 61-68, July 1988.
- [9] E. McDermott, H. Iwamida, S. Katagiri and Y. Tohkura, "Shift-tolerant LVQ and hybrid LVQ-HMM for phoneme recognition," *Readings in Speech Recognition*, pp. 425-438, Morgan Kaufmann, 1990.
- [10] J. Loncelle, N. Derycke and F. F. Soulie, "Cooperation of GBP and LVQ networks for optical character recognition," Int'l Joint Conf. Neural Networks, vol. III, pp. 694-699, June 1992.
- [11] J. Orlando, R. Mann and S. Haykin, "Radar classification of sea-ice using traditional and neural classifiers," Int'l Joint Conf. Neural Networks, vol. II, pp. 263-266, June 1990.
- [12] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, Vol. 1: Foundations, MIT Press, Cambridge, 1986.
- [13] T. D. Knoll and J. T. Lo, "Push-and-pull for piecewise linear Machine training," Int'l Joint Conf. Neural Networks, vol. III, pp. 573-578, June 1992.
- [14] T. Kohonen, J. Kangas, J. Laaksonen and K. Torkkola, *The Learning Vector Quantization program package*, Helsinki Univ. of Technology Lab. of Computer and Information Science, Oct. 1992.
- [15] R. P. Lippmann, "An introduction to computing with neural nets," IEEE ASSP Magazine, vol. 3, no. 4, pp. 4-22, April 1987.
- [16] H. K. Kwan, "One-layer feedforward neural network for fast maximum/minimum determination," *Electronic Letters*, vol. 28, no. 17, pp. 1583-1585, August 1992.

— 著 者 紹 介 —



金 伯 燮(正會員)

1978年 한양대학교 전자공학과 졸업. 1980
年 한국과학기술원 산업전자공학과 졸업(석사).
1985年 한국과학기술원 전기및 전자공학과
졸업(박사). 1984年 ~ 1987年 한양대학교
의과대학 계량의학교실 조교수. 1989年 ~
1990年 미국 Syracuse대학 CASE Center
객원 연구원. 1987年 ~ 현재 한림대학교 전자계산학과 부교
수. 주관심 분야는 패턴인식, 인공신경망, 문자인식, 인공지능
등임.