LVQ를 이용한 퍼지 규칙 생성

이남일, 장광규,신웅철
안동대학교 컴퓨터공학과

Fuzzy Rules Generation Using the LVQ

Lee Nam-il, Jang Kwang-gyu, Shin Woong-cheal
Department of Computer Engineering, Andong National University

요 약

본 논문에서는 Kohonen SOM을 이용한 인식 학습 알고리즘인 LVQ를 이용하여 퍼지 규칙의 수를 줄이는 방안을 제안하였다. 많은 훈련 패턴을 입력하게 되면 그에 따른 퍼지 규칙 수가 증가하게 되고, 많은 기억용량과 분류에 긴 시간을 필요로 하는 문제점 있어 퍼지 규칙의 수를 줄이고자 한다. 그러나 퍼지 규칙의 수가 줄어듦으로서 발생하는 성능의 하락을 최소화하기 위하여 초기 참조 패턴이 입력 데이터에 근접하도록 훈련 된 후에 퍼지 규칙을 생성하였다. 생성된 퍼지 규칙은 LVQ를 이용하여 인식되기 바로 전에 가중치 벡터를 이용하여 근접하는 값 이내에 있는 가중치 벡터 값을 합하여 같은 퍼지 규칙을 부여하여 생성하였다. 그 결과로 5×8 숫자 Gray scale를 이용하여 전체 146개의 가중치 벡터가 15개의 아주 적은 수의 퍼지 규칙으로 생성되었다.

I. 개요

퍼지는 우리가 일상생활에서 나타나는 수치적인 표현으로, 정확하게 표현하기 어려운 애매한 것에 대하여 수치적인 표현이 가능하게 하는 것으로 1965년 자데 교수에 의해 처음으로 제안되었다.[1]

그 이론은 자동차의 속도와 가속기에 대한 수치로 나타내는 분야 및 자동차의 자동운행 분야에서 이용이 되어왔으며, 패턴인식 분야에서도 특징의 선택 및 분류분야에 이용되고 있다. 그러나 아직도 해결되지 못한 문제 중에 퍼지 영역과 Membership Function의 선택이 남아있고, 많은 훈련 패턴을 입력하게 되면 그에 따른 퍼지 규칙 집합 수가 증가하게 되고 기억용량도 더욱 많이 요구되며 분류수행에도 긴 시간을 필요로 하는 문제점이 있다. 뿐만아니라 소량의 데이터라 할지라도 종류별, Pixel별로 Fuzzy Rule을 생성한다면 이것 역시 퍼지 규칙의 수가 많아져서 문제점으로 대두되고 있다.

본 논문에서는 퍼지 규칙의 수를 줄여서 기억용량 문제와 계산에 소요되는 시간을 줄여보고자 한다. 퍼지 규칙의 생성은 LVQ의 초기 참조 모델 훈련을 통하여 입력 데이터의 부류 분포에 근접하도록 갱신시킨 자료를 이용하여[2] 근접하는 값들을 합하여 하나의 퍼지 규칙으로 생성하여 규칙의 수를 줄여나갔다. 본 논문에서는 검증 된 LVQ를 이용한 퍼지 규칙 생성 안을 제안하였다.

본 논문에서 제2장은 SOM에 대한 개요와 알고리즘과 LVQ1 알고리즘에 관하여 알아보고, 제3장에서는 퍼지에 대한 개괄적인 설명과 LVQ 알고리즘을 이용하여 적은 수의 퍼지 규칙을 생성하는 절차 및 방법에 대하여 설명을 하고, 마지막 제4장에서 결론을 기술하였다.

Ⅱ. SOM(Self Organizing Feature Maps)

Kohonen의 Self-Organizing Map(SOM)[3]에서 Self-Organizing이란 주어진 입력 패턴에 대하여 정확한 해답을 미리 주지 않고 스스로 학습할 수 있는 능력을 말한다. 알고리즘은 공통적인 입력노드로부터 M 개의 출력노드로의 연결 가중치를 조정함에 의하여 벡터 계량기(Vector Quantizer)를 만드는 것이다. 연속성 입력벡터를 충분히 제시하면 가중치는 클러스터나 벡터 중심을 지정하게 된다. 벡터 중심의 정밀도 함수는 입력벡터의 확률 밀도 함수를 근사 시킨다. 위상학적으로 가까운 거리의 노드들이 물리적으로 비슷한 입력에 대하여는 매우 민감한 반응을 일으키도록 가중치는 조정된다. 출력 노드는 자연스러운 방법으로 정렬된다.

이렇게 이웃노드는 처음에는 많은 것으로 정의되지만 시간이 지남에 따라 줄어든다.

시간변수가 $0<t1<t2$ 라면

$NEj(t2)<NEj(t1)<NEj(0)$이 다.

이것은 많은 계층의 처리망을 보유하는 복잡한 시스템에서는 매우 중요하다. 그 까닭은 층간 연결 수를 줄일 수 있기 때문이다.

1단계: 가중치초기화

N개의 입력과 M개의 출력노드를 연결하는 연결선에 가중치를 정하되 작은 무작위 수로 한다.

2단계: 새로운 입력을 제시한다.

3단계: 모든 노드들에 대한거리를 산출한다.

입력과 출력노드 j 사이의 거리 dj를 산출 한다. dj 는

$$dj=\sum_{i=0}^{N-1}(Xi(t)-Wij(t))^2$$

과 같다. $Xi(t)$는 시간 t에서 i번째 노드의 입력이다. $Wij(t)$는 시간 t일 때 입력 노드 i로부터 출력노드 j 사이의 가중치를 말한다.

4단계: 최소거리인 출력노드를 선택한다.

최소거리 di를 가지는 노드 $j*$를 선택한다.

5단계: 노드 $j*$와 이웃지간의 가중치를 개정한다.

개정되는 가중치는

$Wij(t+1)=Wij(t)+\eta(t)(Xi(t)-Wij(t))$

로서 모든 $j\in Nj*(t), 0\le i\le N-1$ 에 대하여 매긴다.

$\eta(t)$는 이득항 (gain term)이며

$0<\eta(t)<1$ 이고 시간이 지남에 따라 줄어든다.

6단계: 2단계로 다시 가서 반복한다.

1. LVQ 알고리즘 소개

LVQ 알고리즘은 표본패턴 자체를 참조 모델로 사용하지 않고, 초기 참조 모델을 적절하게 선택한 후에 훈련을 통하여 입력 데이터의 부류 분포에 근접하도록 갱신시킨다. 따라서 각 참조 패턴의 초기값을 어떻게 설정하느냐가 대단히 중요하다고 본다. 현재의 LVQ알고리즘은 LVQ2, LVQ3등 개선된 알고리즘이 있으나 여기서는 LVQ1를 기준으로 설명을 하였다.[4]

2. LVQ1 알고리즘

M개의 영역으로 구성된 표본 공간상에서 각 부류의 참조 패턴이 해당부류의 표본 패턴들이 이루는 군집의 중심에 위치한다고 가정하자. 일반적으로 한 부류에 여러 개의 참조패턴을 배정할 수 있다. 미지의 입력 패턴 x는 참조 모델 내에서 최 근접 거리에 위치한 참조 패턴의 소속 부류와 같은 부류에 해당하는 것으로 결정될 수 있다. 입력 패턴 x와 최 근접 거리에 위치한 참조패턴은 식 (1)에 의해서 결정되는데, 이렇게 결정된 참조 패턴을 m_c로 표기하자.

$$c=\arg\min\{\|x-m_i\|\}.....(1)$$

식 (1)을 사용한 최 근접 이웃 분류에서 오분류율을 최소화시키는 참조패턴 mi는 훈련 데이터

집합을 사용한 훈련을 통하여 접근적으로 결정될 수 있다. x(t)를 시각 t에서의 참조 패턴이라 하자. 초기 참조 모델을 적절하게 선택한 후의 LVQ1 알고리즘은[5] 다음과 같다.

$m_c(t+1) = m_c(t)+\alpha(t)[x(t)-m_c(t)]$
: x와 m_c가 같은 부류에 속할 경우,
$m_c(t+1) = m_c(t)+\alpha(t)[x(t)-m_c(t)]$
: x와 m_c가 서로 다른 부류에 속할 경우

$mi(t+1) = mi(t)$, m_c 이외의 모든 참조 패턴에 대하여,
여기서, $\alpha(t)$는 학습율을 나타내는 상수로 $0<\alpha(t)<1$의 범위 내에서 일정한 값을 갖거나 시간이 경과함에 따라 단조 감소한다. 모든 참조 모델에 각각의 학습율을 배정함으로서 알고리즘이 빠르게 수렴하도록 해줄 수 있다.

III. 퍼지 규칙 생성

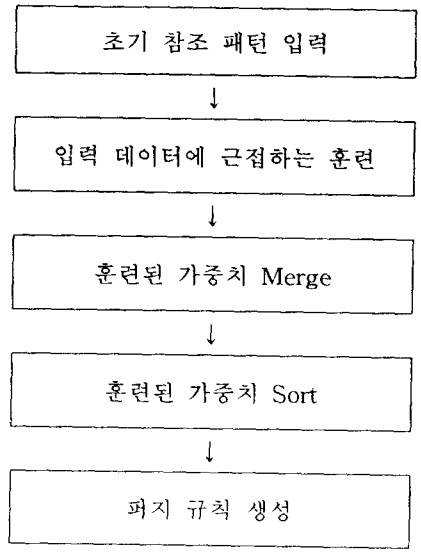
퍼지 규칙은 조건부와 결론부로 구성되며 사용하는 퍼지 규칙의 형태는 다음과 같다.

IF x1 is R1 and x2 is R2 and ··· xn is Rn
THEN y is A

즉, 다수 개의 퍼지 집합으로 구성되어 있는 조건부와 하나의 결론부를 갖는다. 조건부와 결론부는 퍼지 변수(fuzzy variable)와 퍼지값(fuzzy value)들로 구성되며 결론부는 숫자 가능성으로 표현된다.[6]

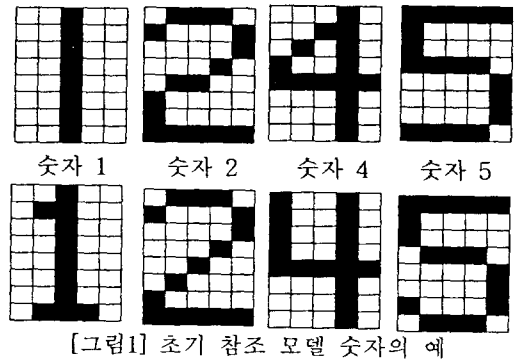
퍼지 규칙의 생성은 LVQ 알고리즘을 훈련과 인식과정 중에 훈련이 끝난 후에 이루어지며, 5 × 8 gray scale 숫자 자료를 사용하였다.
최종의 퍼지 규칙에 나타나는 값은 각각 숫자가 가져야 될 명암값을 나타내게 되며, 퍼지 규칙 생성 과정은 [표 1]과 같이 LVQ의 초기참조 패턴 입력단계와 훈련단계와 퍼지 규칙을 생성하기 위해 3단계로 구성하였다.

[표 1] 퍼지 규칙 생성 과정



1. LVQ에 초기패턴 입력

LVQ 에 입력패턴들의 특징을 충분히 반영할 수 있는 패턴을 선정하여 입력으로 준다.
입력에 필요한 자료 즉 입력패턴의 수, 한 패턴의 크기 , 입력패턴의 종류 입력하는데 초기 참조 모델은 각 숫자가 여러 가지 중요한 특징을 포함하는 특별한 경우를 포함하여야한다.[7]



[그림1] 초기 참조 모델 숫자의 예

2. LVQ 입력 데이타 분포에 근접하게 훈련

LVQ 훈련 알고리즘에 의해 입력데이타 부류 분포에 대하여 모든 경우의 수를 훈련을 통하여 vector값을 변화시킨다.

1.000000	0.000000	0.000000	1.000000	0.000000
1.000000	0.000000	0.500877	1.000000	0.000000
1.000000	0.500377	0.000000	1.000000	0.000000
1.000000	0.000000	0.000000	1.000000	0.000000
1.000000	1.000000	1.000000	1.000000	1.000000
0.000000	0.000000	0.000000	1.000000	0.000000
0.000000	0.000000	0.000000	1.000000	0.000000
0.000000	0.000000	0.000000	1.000000	0.000000

0.000000	0.000000	1.000000	0.000000	0.000000
0.000000	0.500880	1.000000	0.000000	0.000000
0.000000	-0.000000	1.000000	0.000000	0.000000
0.000000	-0.000000	1.000000	0.000000	0.000000
0.000000	-0.000000	1.000000	0.000000	0.000000
0.000000	-0.000000	1.000000	0.000000	0.000000
0.000000	-0.000000	1.000000	0.000000	0.000000
0.000000	0.500877	1.000000	0.500877	0.000000

위의 예와 같이 나타난다.(지면관계상 소수점 6 자리까지만 나타내었다.)

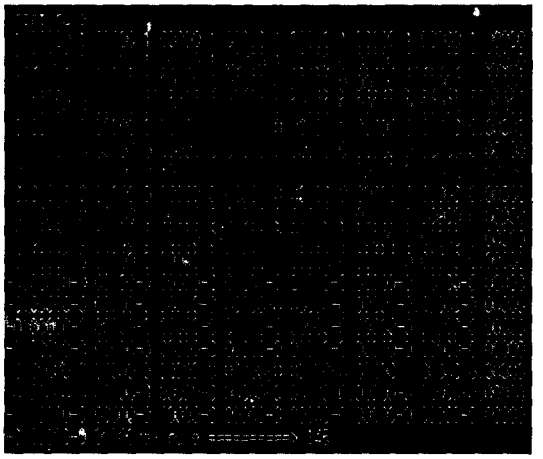
3. 훈련된 가중치 병합

Ⅲ-2에서 만들어진 가중치를 전부 같은 배열에 저장한다. 가중치는 명암을 나타내는 같은 요소를 나타내므로 퍼지 규칙을 생성을 처리하는 동안과 생성하는 중에는 어느 숫자에서 나온 가중치인지는 구분하지 않는다. 최종에 퍼지 규칙을 생성 할 때에 구분은 이루어진다. 퍼지 규칙의 숫자가 줄어드는데 결정적인 역할을 수행하는 단계이다..

4. 훈련된 가중치 정렬

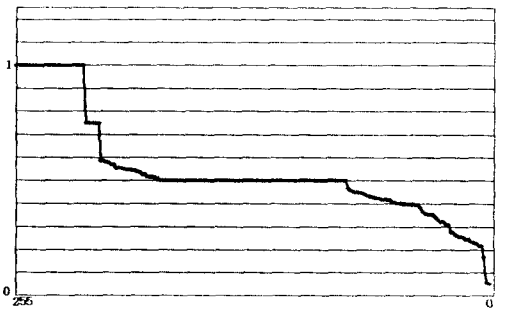
Ⅲ-3에서 병합된 자료를 내림차순으로 정렬을 하는데 이는 다음 단계에서 인접하는 퍼지 규칙에 같은 규칙을 부여하기 위함이다.

특히, 본 논문에서와 같이 단순한 요소 즉, 명암도와 같은 처리에 아주 효율적이며, [그림 2]는 훈련된 가중치 값을 0-9까지 병합한 것을 내림차순으로 정렬한 것이다.



[그림 2] 명암이 높은 순서부터 정렬

[그림 3]은 [그림 2]에 나타난 가중치를 일목요연하게 확인이 가능하게 도식화 한 것이다.



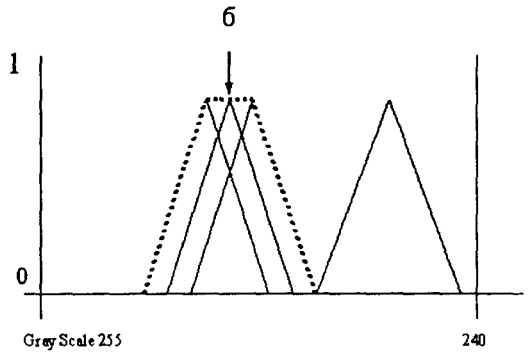
[그림 3] 병합된 200개의 훈련된 가중치 정렬

5. 퍼지 규칙 생성.

정렬된 가중치를 수식(1)에 따라 규칙을 부여하는데 [그림4]는 이를 도식화 한 것으로 점선 부분이 (threshold(б)값을 0.1로 정하였다)하나의 룰이 생성됨을 알 수 있다.[4]

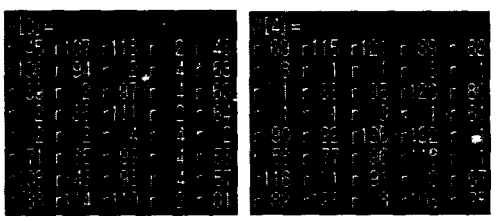
$$\left.\begin{array}{l} | \; w(i) - w(i+1) \; | > б \\ \qquad : \text{Then Assign A New Rule} \\ | \; w(i) - w(i+1) \; | < б \\ \qquad : \text{Then Retain The Rule} \end{array}\right\} \quad ...(1)$$

이 처리를 연속된 영역이 처리 될 때까지 계속해서 처리한다. 따라서 많은 퍼지 영역은 하나의 삼각형 대신에 부등변 사각형을 가진다.

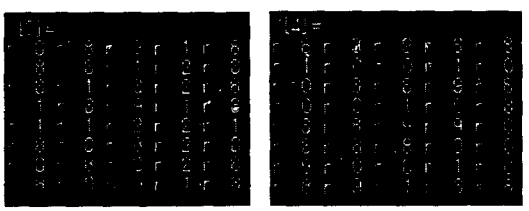


[그림 4] 퍼지 규칙 생성의 예

[그림5]는 본 논문에서 제안한 퍼지 규칙을 적용하지 않은 결과이고, [그림6]은 본 논문에서 제안한 퍼지 규칙을 적용하여 규칙 수가 줄어든 결과를 보여준 것이다.



[그림5] 퍼지 규칙 수를 줄이기 전의 경우



[그림6] 퍼지 규칙의 수가 줄어든 경우

아래 규칙은 위 [그림6]에서 나타난 숫자 "4"에 대한 결과를 퍼지 규칙 IF ~THEN ~문장으로 표현한 것으로 그 결과 퍼지 규칙이 9개임을 알 수 있다.

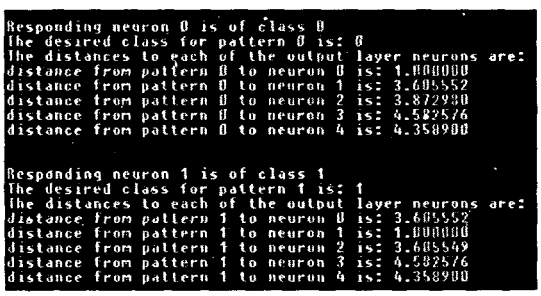
IF

F1 is r8 and F2 is r8 and F3 is r9 and
F4 is r1 and F5 is r8 and F6 is r8 and
F7 is r8 and F8 is r1 and F9 is r2 and
F10 is r8 and F11 is r8 and F12 is r1 and
F13 is r8 and F14 is r2 and F15 is r8 and
F16 is r1 and F17 is r8 and F18 is r9 and
F19 is r1 and F20 is r8 and F21 is r1 and
F22 is r1 and F23 is r2 and F24 is r2 and
F25 is r1 and F26 is r8 and F27 is r8 and
F28 is r8 and F29 is r2 and F30 is r8 and
F31 is r8 and F32 is r8 and F33 is r8 and
F34 is r2 and F35 is r8 and F36 is r8 and
F37 is r9 and F38 is r9 and F39 is r1 and
F40 is r1

THEN

it is digit 4

6. LVQ 알고리즘을 사용한 검증



[그림 7] LVQ를 이용한 인식에 대한 검증

[그림7]에서 생성된 퍼지 규칙이 실제의 인식 작업시 사용이 가능함을 알아보고자 할 때 LVQ 알고리즘의 인식단계를 수행하여 자료의 신빙성을 높이고자 함이다. 다른 퍼지 규칙의 생성은 그 자료에 대한 신빙성의 규명이 어려운 경우가 자주 발생하였다.

4. 결론

본 논문에서는 LVQ를 이용한 퍼지 규칙 생성을 제안하였다. 퍼지 규칙 생성은 LVQ의 초기 참조 모델훈련을 통하여 입력데이터의 부류 분포에 근접하도록 갱신시킨 자료를 이용하여, 근접하는 값들을 합하여 하나의 퍼지 규칙을 생성하여 규칙의 수를 줄여 나갔다.

이는 많은 훈련 패턴 사용 시 발생되는 퍼지 규칙 집합수의 증가와 이에 따른 기억용량의 과대한 요구를 해결하였고, 수행에 소요되는 시간이 절약되었다.

퍼지 규칙의 수는 실험에서 전체 가중치 146개를 인접하는 가중치에 같은 규칙을 생성하여 15개의 적은 퍼지 규칙이 생성되었으며 인접하는 가중치의 값을 적절하게 조정하여 다양한 패턴인식에 필요한 규칙들이 생성 될 것이다..

따라서 이러한 퍼지 규칙은 패턴인식 분야에 많은 참조가 될 것으로 기대되며, 향후 과제는 변화가 많은 응용분야와 학습량이 많은 지문인식, 얼굴인식분야에 적용하는 것이다.

참고문헌

[1] Zadeh,L.A."Fuzzy sets.Information Control", pp338~353, 1965. 8.

[2] Zhejru Chi and Hong Yan,"Handwritten Numeral Recognition Using a Small Number of Fuzzy Rules With Optimezed Defuzzification Parameters", The Hong Kong Polytechnic University and The University of Sydney, Nov., 1994

[3] 김희승,"영상인식-영상처리,컴퓨터 비젼,패턴인식,신경망-",생능출판사, pp343-346, 1997.8

[4] 송시헌,이성환,"개선된LVQ3와 시뮬레이티드 아닐리의 결합을 통한 대용량 필기체 문자 인식을 위한 최적의 참조 모델 설계",정보과학회 논문지 제21권 제6호, pp59-1607, 1994. 6

[5] Abhijit S. Pandya & Rovert B. Macy, Pattern Recognition with Neural Networks in C++ ,CRC Press, 1996

[6] 菅野道夫저, 朴玟用, 崔恒植역, "퍼지 제어 시스템", 대영사, pp70~135, 1990. 9.

[7] Dickerson, J.,& Kosko. B. Fuzzy function learning with covariance ellipsoids. Proceedings of IEEE International Conference on Neural Networks,Ⅲ,1162-1167, San Francisco, CA, 1993.