

# 자기조직화 특성지도와 퍼지근사추론을 결합한 개선된 형태의 퍼지근사추론에 관한 연구\*

이건창\*\* · 조형래\*\*\*

## An Improved Method of Method of Fuzzy Approximate Reasoning by Combining Self-Organizing Feature Map and Fuzzy Logic\*

Kun Chang Lee\*\* · Hyung Rae Cho\*\*\*

### Abstract

This paper proposes a new type of fuzzy approximate reasoning method that combines a self organizing feature map and a fuzzy logic. Previous methods considered only input part to determine the number of fuzzy rules, while this paper considers both input and output parts simultaneously. Our approach proved to improve the inference performance. We also developed a new index for avoiding overlearning which guarantees more accurate results. Experimental results showed that our approach surpasses the performance of Takagi & Hayashi(1991) approach.

## 1. 서론

주어진 문제에 대하여 전문가가 추론하는 방식은 대개 전문적 지식에 기초한 근사추론 (approximate reasoning) 방법을 사용한다. 왜냐

하면,

첫째, 전문가가 해결하여야 하는 문제는 대개 비정형화된 (unstructured) 문제가 대부분이어서, 일정한 풀이과정이 존재하지 않기 때문이다.

둘째, 주어진 문제에 대하여 불확실한 (fuzzy) 일부의 정보를 (incomplete

\* 이 논문은 경상대학교 생산기술연구소의 연구장려금으로 수행되었음.

\*\* 성균관대학교 경영학부 교수

\*\*\* 경상대학교 산업공학과 교수

information) 기초로 하여 정확한 추론을 하여야 하는 경우가 일반적이기 때문이다.

이 경우 행하는 추론방식이 전문가가 가지고 있는 해당 분야의 전문가적 지식에 기초한 근사 추론 방식이다. 인공지능 방법에서는 이와 유사한 기법이 전문가시스템 (expert system), 퍼지이론 (fuzzy logic theory), 그리고 인공신경망 (neural network) 이론이 있다. 이중 전문가시스템은 해당 문제영역에 대한 전문적 지식베이스를 기초로 하여 추론하는 방식이다. 전문가시스템에서 사용하는 지식의 형태는 IF-THEN 규칙이 가장 널리 사용된다 (Buchanan & Shortliffe 1984; Giarratano & Riley 1994; Silverman 1987; Waterman 1986). 그러나, 이러한 전문가시스템이 근사추론 방식에 적합하려면 해당 지식의 형태가 퍼지화되어야 한다. 즉, 각 지식이 불확실하고 불완전한 일부의 정보만을 가지고 추론이 가능하도록 구성이 되어 있어야 한다는 것이다. 이렇게 퍼지화된 지식을 사용하는 전문가시스템이 퍼지전문가시스템 (fuzzy expert system)이다 (Graham 1991; Leung et al. 1989; Negoita 1985). 한편, 퍼지이론의 경우 Zadeh교수가 제안한 가능성이론 (possibility theory)이론에 기초하여 불확실한 정보를 처리하는 기법이다. 이는 해당 변수에 대한 퍼지멤버십 함수 (fuzzy membership function)를 정의하여 변수관련 정보에 숨겨져 있는 불확실성이나 불완전성을 논리적이고 체계적으로 처리하는 방법이다 (Zadeh 1973, 1983). 인공신경망 이론은 이미 널리 알려진 바와 같이 주어진 자료에 내재되어 있는 정형성

(regularities)을 학습하는 학습능력과, 새로운 자료에 대하여 추론할 수 있는 일반화 (generalization) 능력이 우수하다 (Rumelhart et al. 1986).

이들 각각의 인공지능 근사추론 방법은 서로 장단점이 있다. 따라서, 이들 방법을 서로 결합하여 사용한다면 어느 한가지 방법을 사용하는 경우보다 그 성과가 더 크다. 본 논문에서는 특히 인공신경망과 퍼지이론을 결합한 새로운 방식의 퍼지근사추론 방식을 제시하려고 한다. 인공신경망 방법은 특히 뛰어난 학습기능과 일반화 능력<sup>1)</sup> 때문에 비정형화된 문제영역에 적합한 특성을 갖고 있다. 그러나, 인공신경망은 불확실한 정보를 처리하는 체계적인 메카니즘이 결여되어 있다는 단점이 있는 반면에, 퍼지이론은 일반화 효과와 학습능력은 없지만 불확실한 정보를 처리하는 메카니즘을 가지고 있다.

본 연구에서 주제로 하고 있는 인공신경망과 퍼지로직의 결합을 통한 퍼지근사추론 방법에 관한 연구는 많은 학자들에 의하여 최근 매우 중요한 연구주제로 다뤄지고 있다. Jang (1992)은 퍼지근사추론 시스템을 인공신경망으로 전환하는 방법을 제시하고 해당 인공신경망을 역전파학습 방법으로 학습시키므로써 해당 파라미터를 조정해가는 방법을 제안하였다. 기타 몇몇 연구자들도 Jang (1992)과 비슷한 방법을 제안하였다 (Fukuda & Shibata, 1992; Shibata et al, 1992; Wang & Mendel, 1992). Nakayama et al (1992)은 퍼지근사추론을 위한 특수한 구조를 갖는 퍼지인공신경망을 제안하였고, Takagi & Hayashi (1991)는 퍼지근사추론을

1) 일반화 효과란 학습하지 않은 입력자료에 대하여 추론할 수 있는 능력을 의미한다. 따라서 전문가가 가지고 있는 근사추론 능력과 가장 유사한 것이 바로 이 일반화 효과라고 할 수 있다.

위하여 역전파학습 인공신경망을 적용하여 이를 제어문제에 적용하였다 (Takagi & Hayashi, 1991). 기존의 인공신경망과 퍼지근사추론과의 결합은 주로 역전파학습 인공신경망 모형에 의존하기 때문에, 추론시 IF부분의 확인과 THEN 부분 확인을 위하여 많은 학습이 요구되었다. 또한, THEN부분을 구성하는 변수중 추론성공에 크게 공헌을 하지 않는 변수를 가려내기 위하여 반복적인 학습과정을 통한 변수제거법을 사용하나, 이것 역시 많은 학습이 요구된다는 문제가 있었다. 이러한 문제점은 특히 입력변수가 많아 상당한 학습시간이 요구되는 대규모 실제문제에서 더욱 두드러져 실제응용에 문제가 있을 수가 있다.

그러나, 본 연구에서는 이러한 기존의 인공신경망과 퍼지논리를 결합한 퍼지근사추론 방법의 문제점을 개선한 새로운 형태의 퍼지근사추론 방법을 제안한다. 본 연구에서 제안하는 퍼지근사추론 방법의 기본 아이디어를 정리하면 다음과 같다.

- (1) 비감독학습 인공신경망 모형인 SOFM (Kohonen, 1990)의 결과인 클러스터(cluster)를 이용하여, IF-THEN규칙중 적정한 IF부분과 THEN부분을 찾아낸다.
- (2) 이때, 적정한 퍼지 IF-THEN규칙의 수를 결정함에 있어서, 입력 클러스터 뿐만 아니라 출력 클러스터도 동시에 고려한다.
- (3) 각 퍼지규칙을 학습함에 있어서 과잉학습(overlearning) 방지를 위한 새로운 인덱스를 개발한다.

본 연구는 다음과 같은 형태로 구성되어 있다. 우선, 2장에서는 퍼지근사추론의 특성에 대

하여 소개하고, 그 장단점을 언급한 후, 3장에서 인공신경망과 결합할 수 있는 가능성을 Takagi & Hayashi (1991)의 연구를 중심으로 소개한다. 그리고 본 연구에서 제안하는 새로운 형태의 퍼지근사추론 메카니즘의 내용을 4장에서 소개하고, 아울러 5장에서 실험결과를 소개한다. 6장에서 결론 및 향후 연구방향을 소개하고 이 논문을 끝맺는다.

## 2. 퍼지근사추론과 인공신경망

다음과 같은 전통적인 크리쉬(crisp) 추론규칙을 보자.

IF       회전각도 = 20° - 40° and  
          운반중량 = 50kg 이상,  
THEN    운행속도 = 3 - 회전각도/20.

이러한 크리쉬 규칙의 문제점은 운반중량이 49.9999 kg인 경우에, 이같은 추론규칙을 적용할 수 없다는 것이다. 더욱이, 회전각도가 20 ± α° 일 경우에 두 개 이상의 제어규칙이 불규칙하게 임의로 적용되므로 제어가 부드럽게 이뤄지지 못한다는 것이다. 반면에, 이에 상응하는 퍼지근사추론 규칙은 다음과 같이 표현할 수 있다.

IF       회전각도 = 적정한( moderate)각도 and  
          운반중량 = 무겁다(heavy),  
THEN    운행속도 = 점차(gradually) 감소.

이러한 퍼지근사추론 규칙은 인간의 추론방식과 유사하여 이해하기 쉬울뿐만 아니라, 추론

에 적용되는 규칙의 개수가 크게 감소된다는 장점이 있다. 그러나 이같은 퍼지근사추론을 이용하기 위해서는 다음과 같은 문제들이 해결되어야 한다.

첫째, 주어진 문제영역에 필요한 퍼지 규칙의 수는 어떻게 구할 수 있는가? 퍼지 규칙이란 기본적으로 조건부분(IF 부분)에 해당하는 퍼지집합과 결론부분(THEN 부분)에 해당하는 퍼지집합간의 관계라고 정의할 수 있다. 따라서 필요한 퍼지규칙의 수를 결정하기 위해서는 IF부분을 위한 퍼지집합과 THEN부분을 위한 퍼지집합의 도출이 선행되어야 한다. 문제영역에 따라 IF 또는 THEN부분에 필요한 퍼지집합이 주어지는 경우도 있다. 하지만 앞서 예들 든 제어문제등의 경우에는 센서(sensor)에 의해 획득된 수치 자료를 바탕으로 "적정한", "무겁다" 등과 같은 규칙의 조건부분을 위한 퍼지집합과, "점차 감소" 등과 같은 결론부분을 위한 퍼지집합을 도출해야 한다.

둘째, 규칙의 IF부분에 속하는 퍼지집합에 대한 멤버쉽함수(membership function)는 어떻게 설계되어야 하는가? 가장 널리 사용되는 퍼지 멤버쉽함수의 형태는 삼각형 형태의(triangular) 함수이거나, 또는 사다리꼴 형태의(trapezoidal) 함수이다. 그러나 주어진 문제영역의 특성에 따라 이같은 퍼지 멤버쉽함수의 형태는 가변적으로 될 수 밖에 없다. 특히 주어진 퍼지근사추론에서 고려해야할 변수가 늘어나면 단순히 의사결정자의 경험과 직관을 통한 주관적 판단만으로는 멤버쉽 함수를 설계하는 것이 거의 불가능할 정도로 복잡해 진다는 문제점이 있다. 이것이 퍼지근사추론에서 극복해야할 두 번째 문제점이다.

세 번째 문제점은 퍼지근사추론에는 주어진

환경의 변화에 대한 자체적인 학습기능이 결여되어 있다는 점이다. 결국 이와같이 퍼지추론에 있어서의 문제점을 해결할 수 있는 대안으로서 기존의 학자들은 인공신경망과 퍼지근사추론의 결합을 적극적으로 모색하여 왔다 (Fukuda & Shibata, 1992; Funahashi, 1989; Jang 1992; Nakayama et al, 1992; Shibata et al, 1992; Takagi & Hayashi, 1988, 1991; Wang & Mendel, 1992).

한편, 인공신경망의 수학적인 특징은 이미 McCulloch & Pitts(1943)과 Rosenblatt(1958)에 의하여 제시되어 왔으나, 이러한 인공신경망이 본격적으로 의사결정분야나 기타 공학분야에 본격적으로 적용된 것은 Rumelhart et al(1986)이 제시한 역전파 학습 알고리즘(Backpropagation Learning Algorithm)이 등장한 이후가 된다. 이같은 인공신경망은 퍼지근사추론과 상호보완적인 특성을 가지고 있다. 즉, 퍼지근사추론의 경우 지식이 규칙형태로 표현됨으로써 해당 추론 과정이 보다 분명하게 드러나는 반면에, 인공신경망의 경우는 추론과정이 인공신경망 자체내의 여러 뉴런(neuron) 사이의 연결가중치(connection weights)에 의하여 비선형적으로 결정되므로 추론과정이 분명하게 제시되지 않는다. 그러나, 퍼지근사추론의 첫 번째 문제점인 퍼지집합의 도출과 두 번째 문제점인 퍼지멤버쉽함수를 주어진 자료를 바탕으로 인공신경망에 의하여 결정되게 한다면, 해당 퍼지집합 및 퍼지멤버쉽함수 결정을 위한 자의성이 어느 정도 해결될 수가 있다. 또한, 인공신경망은 문제영역의 자료가 변화되어도 이를 학습기능을 통하여 얼마든지 반영할 수가 있다. 이같이 인공신경망과 퍼지근사추론이 결합되므로써, 앞서 언급한 퍼지근사추론의 세가지 문제점이 해결될 수가 있다.

### 3. 제안된 퍼지근사추론 메카니즘

본 연구에서 제안하는 퍼지근사추론 메카니즘은 크게 준비단계, 퍼지규칙의 수를 결정하는 단계, 퍼지규칙에 대한 멤버십 함수를 구하는 단계, 규칙의 결론부분을 학습하는 단계 및 최종 의사결정단계로 나누어 진다. 이제 각 단계 별로 수행해야 할 내용 및 수행 방법을 기술하면 다음과 같다.

#### STEP 1 : 준비단계

출력을 나타내는 벡터를  $Y = (y_1, \dots, y_m)$ 로 표시하고 출력값에 영향을 미치는 입력변수의 벡터  $X = (x_1, \dots, x_k)$ 를 결정한다.  $(X, Y)$ 에 대한 과거자료를 수집하고,  $(X_i, Y_j), i = 1, \dots, n$ , 수집된 전체  $n$ 개의 자료를  $n_t$ 개의 학습자료와  $n_c$ 개의 테스트 자료로 나눈다,  $n = n_t + n_c$ .

#### STEP 2 : 퍼지규칙의 수 결정

이 단계는 주어진 문제해결을 위해 필요한 퍼지규칙의 수 및 각 퍼지규칙에 속하는 학습자료를 결정하는 단계이다. 일반적인 퍼지규칙은 다음과 같이 표시될 수 있다.

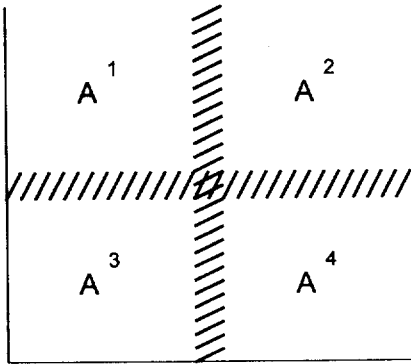
$$\begin{aligned} \text{IF } X \in A_i \\ \text{THEN } Y = u_{ij}(X) \in B_j \end{aligned} \quad (\text{식 1})$$

여기서  $A_i, B_j$ 는 각각 하나의 조건부(IF부분) 및 결론부(THEN부분)에 속하는 퍼지집합을 의미한다. 그리고  $u_{ij}(X)$ 는 퍼지집합

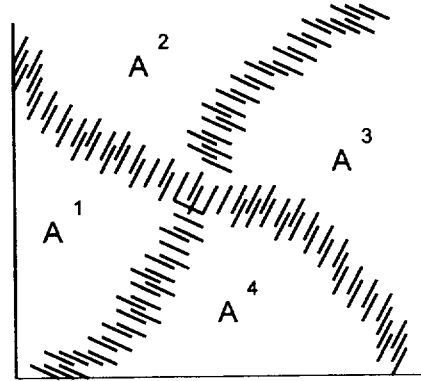
$A_i$ 에 속하는  $X$ 의 값을 퍼지집합  $B_j$ 에 속하는  $Y$ 의 값으로 변환시키는 함수이다. (식 1)에서 보듯이 퍼지규칙이란 기본적으로 IF부분에 속하는 퍼지집합과 THEN부분에 속하는 퍼지집합간의 관계라고 생각할 수 있다. 따라서 문제 해결을 위해 필요로 하는 퍼지규칙의 수를 결정하기 위해서는 우선 학습자료의 입력부분 및 출력부분을 바탕으로 퍼지규칙의 IF부분에 속하는 퍼지집합과 THEN부분에 속하는 퍼지집합을 도출하는 것이 필요하다. 그런데 입력(또는 출력)부분이 2개 이상의 변수로 구성되어 있는 경우에는 벡터에 대한 퍼지집합을 구해야 한다. 기존에는 벡터를 구성하는 변수들이 상호독립이라는 가정하에 각 변수별로 퍼지집합을 구한 후 이를 합성하여 [그림 1-a]와 같은 형태의 벡터에 대한 퍼지집합을 구하였다. 하지만 다음 예와 같이 입출력 벡터를 구성하고 있는 변수들이 상호 의존적인 경우도 많다.

$$\begin{aligned} \text{IF } \text{온도가 조금 높아지고 and} \\ \text{습도가 낮아지면} \\ \text{THEN 동력을 조금 낮추어야 한다.} \end{aligned}$$

여기서 입력을 구성하는 두 변수인 온도와 습도가 상호독립이라고 가정하기 어렵다는 것이다. 이렇게 상호 의존적인 변수들로 구성된 벡터에 대한 퍼지집합의 형태는 [그림 1-b]와 같이 나타난다. 그런데 벡터를 구성하는 변수의 수가 증가할수록 [그림 1-b]와 같은 형태의 퍼지집합 도출이 거의 불가능할 정도로 복잡해진다는데 문제점이 있다.



(a) 상호독립인 변수로 구성된 벡터에 관한 퍼지집합 분할



(b) 일반적인 벡터에 대한 퍼지집합 분할

[그림 1] 벡터에 대한 퍼지집합

본 연구에서는 벡터에 대해 [그림 1-b]와 같은 일반적인 형태의 퍼지집합을 구하기 위하여 비감독학습 인공지능망인 SOFM(Self-Organized Feature Map)을 사용한다. Kohonen(1982)에 의해 개발된 SOFM은 역전파학습 모형과는 달리 학습시 기대출력값이 필요없는 비감독학습방법을 사용한다. SOFM은 '입력층'과 '경쟁층'의 2차원 격자(two-dimensional grid)로 구성되어 있다. 모든 입력은 가변 연결강도를 통하여 모든 출력뉴런에 연결되고 출력뉴런은 많은 국부적 접속으로 상호 연결되어 있으며, 입력으로는 연속값을 갖는 입력자료가 기대출력값이 표기되지 않은 상태로 순차적으로 주어진다. SOFM의 학습시 연결강도 벡터와 입력자료가 얼마나 가까운가를 계산하여 가장 가까운 경쟁층내의 출력뉴런이 활성화된다. 이 활성화된 출력뉴런이 출력신호를 보낼 수 있는 유일한 출력뉴런으로서 선택된다. 한편 이 선택된 출력뉴런과 그의 이웃반경안의 뉴런들은 Kohonen의 학습규칙에 따라 연결강도를 조정한다. 다음은 SOFM의 학습알고리즘을 순차적으로 정리한 것이다.

단계 1. 연결강도를 초기화한다.

[그림 1]과 같이 N개의 입력과 출력뉴런 사이의 연결강도를 임의의 작은수로 초기화한다. 경쟁층내에서 초기의 이웃 반경은 모든 출력뉴런들이 포함될 수 있도록 충분히 크게 잡았다가 점차로 줄여나간다.

단계 2. 새로운 입력자료를 제시한다.

단계 3. 입력자료와 모든 출력뉴런들간의 거리를 계산한다. 입력과 출력뉴런 j사이의 거리  $d_j$ 는 다음과 같이 계산한다.

$$d_j = \sum_{i=1}^N (x_i(t) - w_{ij}(t))^2$$

여기서 t는 반복횟수를 의미한다.  $x_i(t)$ 는 t시점에서의 i번째 입력자료이고  $w_{ij}(t)$ 는 t시점에서의 i번째 입력자료와 j번째 출력뉴런 사이의 연결강도이다.

단계 4. 해당 입력자료와 최소거리에 있는 출력뉴런을 선택한다. 즉, 최소거리  $d_j$ 인 출력뉴런  $j^*$ 를 선택한다.

단계 5. 출력뉴런  $j^*$ 와 그 이웃들의 연결강

도를 재조정한다. 출력뉴런  $j^*$ 와 그 이웃반경내의 뉴런들의 연결강도를 다음식에 의해 재조정한다.

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(x_i(t) - w_{ij}(t))$$

여기서  $j$ 는 출력뉴런  $j^*$ 의 이웃반경내의 출력 뉴런을 의미하고,  $\alpha$ 는 0과 1사이의 값으로서 시간이 경과함에 따라 점차 작아진다.

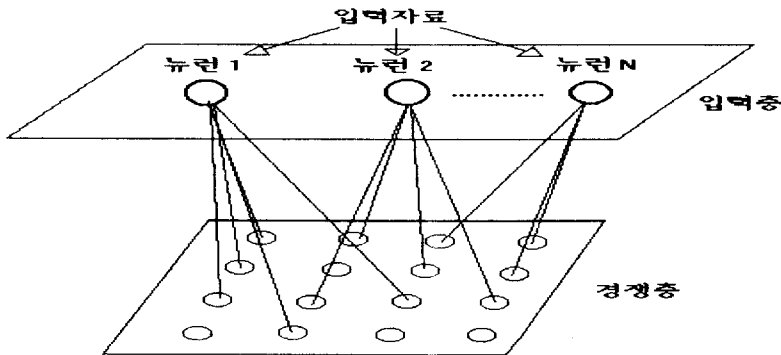
단계 6. 단계 2로 가서 반복한다.

SOFM 학습의 처음 단계에서는 경쟁층내의 모든 출력뉴런들이 포함될 수 있으나, 학습이 진행됨에 따라 점점 적은 수의 출력뉴런들이 학습을 하고 최종적으로는 단지 '승자뉴런'만이 연결강도를 조절하게 된다. 이러한 과정이 끝나면 또 다른 입력자료가 들어오고 이 과정을 되

풀이한다. 이렇게 해서 입력자료중 서로 유사한 자료들끼리 클러스터로 분류된다. [그림 2]는 SOFM의 기본구조를 도시하고 있다.

SOFM의 학습시에 각 뉴런들은 연결강도 벡터와 입력벡터가 얼마나 가까운가를 계산하여 가장 가까운 뉴런을 활성화시킨다. 그러므로 SOFM에서 나온 결과값은 입력자료들을 서로 유사한 값들로 분류해 준다. 본 연구에서는 SOFM에 의해 학습자료의 입력(또는 출력)부분을 분류하고 분류된 각각의 클러스터(Cluster)를 IF(또는 THEN)부분의 퍼지집합으로 간주한다. SOFM에 의한 분류 결과는 다음과 같이 네 개의 경우로 나누어 생각해 볼 수 있다.

이제 [표 1]의 각 경우에 대해서 퍼지규칙의 수 및 각 퍼지규칙에 해당하는 학습자료를 도출



[그림 2] SOFM의 기본구조

[표 1] SOFM에 의한 분류결과

구 분		입력부분	
		클러스터링 가능	클러스터링 불가능
출 력 부 분	클러스터링 가능	경우 1 : $p > 1, q > 1$	경우 3 : $p = 1, q > 1$
	클러스터링 불가능	경우 2 : $p > 1, q = 1$	경우 4 : $p = 1, q = 1$

하는 과정은 다음과 같다. 이때  $p$ 는 IF부분 퍼지집합의 수를 의미하고,  $q$ 는 THEN부분 퍼지집합의 수를 나타낸다. 특히 [표 1]에서 주지할 사항은  $p$ 나  $q$ 가 1이면 클러스터링이 안되지만 2개 이상이면 클러스터링이 가능하다는 사실이다.

경우 1 : 입출력 모두 클러스터링 가능

이 경우는 SOFM에 의한 분류 결과  $A_1, \dots, A_p$ 라는  $p(>1)$ 개의 IF부분 퍼지집합과  $B_1, \dots, B_q$ 라는  $q(>1)$ 개의 THEN부분 퍼지집합이 있다고 나타난 경우이다. 이때 IF부분 퍼지집합과 THEN부분 퍼지집합의 관계를 바탕으로 문제해결을 위해 필요한 퍼지규칙의 수와 각 퍼지규칙에 해당하는 학습자료를 도출하는 과정은 다음과 같다. 우선 [표 2]와 같이 행렬형태의 퍼지집합 관계표를 작성한다.

[표 2] 퍼지집합 관계표

구분	$A_1$	$A_2$	...	$A_p$
$B_1$			.....	
⋮	⋮	⋮	.....	⋮
$B_q$			.....	

다음으로  $n_t$ 개의 각 학습자료를 입력부분 및 출력부분이 각각 어느 퍼지집합에 속하는가를 바탕으로 퍼지집합 관계표의 해당하는 셀(cell)에 분류한다. 그결과 해당하는 학습자료가 존재하는 퍼지집합 관계표의 각 셀을 하나의 퍼지규칙으로 간주한다. 이 경우 해당하는 학습자료가 없는 셀도 존재할 수 있기 때문에 최종적인 퍼지규칙의 수  $r$ 은 다음과 같다:

$$\max(p, q) \leq r \leq p \times q.$$

경우 2 : 입력은 클러스터링 가능 출력은 불가능

입력이  $p(>1)$ 개의 클러스터로 나누어지는데 반해 출력은 2개 이상의 클러스터로 분리하는데에 실패한 경우이다. 이때는 퍼지집합 관계표가  $p$ 개의 셀로 구성된 행벡터형태가 된다. 경우 1과 같은 방법으로  $n_t$ 개의 학습자료를 해당하는 셀에 분류하면 각 셀마다 최소한 하나 이상의 자료가 존재하게 된다. 따라서 이 경우에는  $p$ 개의 입력 클러스터 수만큼의 퍼지규칙이 존재한다고 생각할 수 있다.  $r = p$ .

경우 3 : 입력은 클러스터링 불가능 출력은 가능

경우 2와는 반대로 입력은 2개 이상의 클러스터로 분리되지 않는 반면에 출력은  $q(>1)$ 개의 클러스터로 나누어진 경우이다. 이때는 퍼지집합 관계표가 다음과 같이  $q$ 개의 셀로 구성된 열벡터 형태가 된다.  $n_t$ 개의 학습자료를 퍼지집합 관계표의 해당하는 셀에 분류하면 각 셀마다 최소한 하나 이상의 자료가 존재하게 된다. 따라서 이 경우에는  $q$ 개의 출력 클러스터 수만큼의 퍼지규칙이 존재한다고 생각할 수 있다.  $r = q$ .

경우 4 : 입출력 모두 클러스터링 불가능

이 경우는 퍼지집합을 도출할 수 있는 아무런 정보도 없다. 다시말해 SOFM에 의해 문제해결을 위한 퍼지규칙의 수를 결정하는 것이 불가능하다는 것이다. 따라서 이 경우에는 기존의 인공지능망을 통한 추론방식을 적용할 수 밖



에 없다.

**STEP 3 : 퍼지규칙에 대한 멤버십 학습**

STEP 1의 결과로서  $R^s, s = 1, \dots, r$ 이라는  $r$ 개의 퍼지규칙과 각 퍼지규칙  $R^s$ 에 해당하는 학습자료  $(X_i^s, Y_i^s), i = 1, \dots, (n_t)^s$ 를 구하였다. STEP 2는 각 퍼지규칙에 대한 멤버십함수를 만드는 인공신경망 NNmem을 생성하는 단계이다. 여기서 퍼지규칙의 멤버십함수란 해당 퍼지규칙의 적용가능성 정도를 의미한다.  $X_j$ 가 입력벡터의 값이라면 NNmem을 통해 원하는 출력값은  $X_j$ 의 각 퍼지규칙  $R^s$ 에 대한 멤버십을 나타내는  $\mu_{R^s}(X_j), s = 1, \dots, r$ 이다. STEP 1의 결과를 바탕으로 이러한 기능을 갖는 NNmem을 생성하기 위해서는 각 학습자료의 입력값  $X_i, i = 1, \dots, n_t$ 에 대해 다음의 출력값  $O_i^s$ 를 갖도록 학습시킨다.

$$O_i^s = \begin{cases} 1, & X_i \in R^s \\ 0, & X_i \notin R^s \end{cases}$$

$$i = 1, \dots, n_t : s = 1, \dots, r$$

이렇게 학습된 NNmem은 임의의 입력값  $X_j$ 에 대해 각 퍼지규칙  $R^s$ 의 적용가능성 정도를 나타내는 멤버십  $\mu_{R^s}(X_j), s = 1, \dots, r$ 를 제공해 준다.

**STEP 4 : 퍼지규칙 학습**

이 단계는 (식 1)과 같은 형태의 각 퍼지규

칙  $R^s$ 별로 결론에 해당하는  $Y = u_s(X)$ 를 추론하기 위한 인공신경망 NNs,  $s = 1, \dots, r$ 를 생성하는 단계이다. NNs를 학습하기 위해서는 퍼지규칙  $R^s$ 에 해당하는 학습자료의 입력부분인  $X_i^s, i = 1, \dots, (n_t)^s$  및 출력부분인  $Y_i^s, i = 1, \dots, (n_t)^s$ 가 각각 NNs의 입력값과 출력값으로 부여된다. 이렇게 학습된 NNs는 규칙  $R^s$ 의 THEN부분 역할을 한다. 그런데 NNs와 같은 인공신경망을 학습하는데 있어서 주의해야 할 사항은 과잉학습(overlearning)을 방지해야 한다는 것이다. 일반적으로 인공신경망을 과잉학습시킬 경우 학습자료에 대한 오차는 줄어들지만 테스트자료에 대한 오차는 증가한다. 이러한 과잉학습의 문제를 방지하기 위해서는 학습자료에 대한 오차뿐만 아니라 테스트자료에 대한 오차도 동시에 고려하는 것이 필요하다. 규칙  $R^s$ 에 해당하는 학습자료  $(X_i^s, Y_i^s), i = 1, \dots, (n_t)^s$ 에 대한 NNs의 추론값을  $u_s(X_i^s), i = 1, \dots, (n_t)^s$ 라 하면 NNs의 학습자료에 대한 오차제곱합  $\Theta_t^s$ 는 다음과 같이 구해진다.

$$\Theta_t^s = \sum_{i=1}^{(n_t)^s} \{Y_i^s - u_s(X_i^s)\}^2 \quad (\text{식 2})$$

또한 각 테스트자료  $(X_j, Y_j)$ 에 대한 NNmem의 출력값을  $\mu_{R^s}(X_j)$ , NNs의 출력값을  $u_s(X_j), j = 1, \dots, n_c; s = 1, \dots, r$ 라 하면 NNs의 테스트자료에 대한 오차제곱합  $\Theta_c^s$ 는 다음과 같이 근사화된다 (부록 참조).

$$\theta_c^s = \sum_{j=1}^{n_c} \left[ \frac{\mu_{R^s}(X_j)}{\sum_{s=1}^r \mu_{R^s}(X_j)} (Y_j - u_s(X_j)) \right]^2 \quad (\text{식 3})$$

이상의  $\theta_c^s$  및  $\theta_c^s$ 를 바탕으로 본 연구에서는 NNs에 대한 과잉학습을 방지하기 위해 다음과 같이  $\theta_c^s$  와  $\theta_c^s$ 의 가중평균인  $I^s$ 의 값이 최소가 될 때까지 학습시켰다.

$$I^s = \frac{n_c}{(n_t)^s + n_c} \sum_{i=1}^{(n_t)^s} \{Y_i^s - u_s(X_i^s)\}^2 + \frac{(n_t)^s}{(n_t)^s + n_c} \sum_{j=1}^{n_c} \left[ \frac{\mu_{R^s}(X_j)}{\sum_{s=1}^r \mu_{R^s}(X_j)} (Y_j - u_s(X_j)) \right]^2 \quad (\text{식 4})$$

기존의 연구에서는 이상과 같은 방식을 통해서 도출된  $\theta_c^s$ 를 바탕으로 퍼지규칙  $R^s$ 의 THEN 부분에서 필요없는 입력변수가 있는지 여부를 검토하였다. 다시말해  $u_s(X)$ 를 추론하는데 있어서 입력벡터를 구성하는 변수중 일부를 제거했을 경우 제거하지 않았을 때보다 더욱 작은  $\theta_c^s$ 를 나타내면 해당 변수들을 제거한다는 것이다. 그런데 본 연구에서는 이러한 변수제거 과정을 생략하였다. 그 이유는 준비단계에서 출력값을 추론하기 위해 필요하다고 선정되었던 변수들이기 때문에 제거를 통해  $\theta_c^s$ 가 줄어드는 경우가 거의 없고, 특히  $\theta_c^s$ 가 이론적인 정확한 수치가 아닌 표본을 바탕으로 계산된 근사치이기 때문에 학습자료 및 테스트자료의 변화에 따라 다른 결과가 나올 수도 있기 때문이다.

**STEP 5 : 의사결정**

이상의 과정을 통해 NNmem 및 NNs,  $s = 1, \dots, r$ 이 학습되면 임의의 입력값  $X_k$ 에 대한 최종 추론값인  $Y_k^*$ 를 구하는 절차는 다음과

같다 ([그림 2] 참조).

- ① NNmem에  $X_k$ 를 입력하여 각 퍼지규칙  $R^s$ 의  $X_k$ 에 대한 적용가능성을 의미하는 멤버십  $\mu_{R^s}(X_k)$ ,  $s = 1, \dots, r$ 을 구한다.
- ② NNs에  $X_k$ 를 입력하여 각 퍼지규칙  $R^s$ 의 추론값  $u_s(X_k)$ ,  $s = 1, \dots, r$ 을 구한다.
- ③ 다음 식에 의해 최종 추론값인  $Y_k^*$ 를 구한다.

$$Y_k^* = \frac{\sum_{s=1}^r \mu_{R^s}(X_k) \cdot u_s(X_k)}{\sum_{s=1}^r \mu_{R^s}(X_k)} \quad (\text{식 5})$$

[그림 3]은 본 논문에서 제안된 퍼지근사추론 메카니즘에 의한 의사결정 절차를 표시하고 있다.

**4. 실험 및 결과**

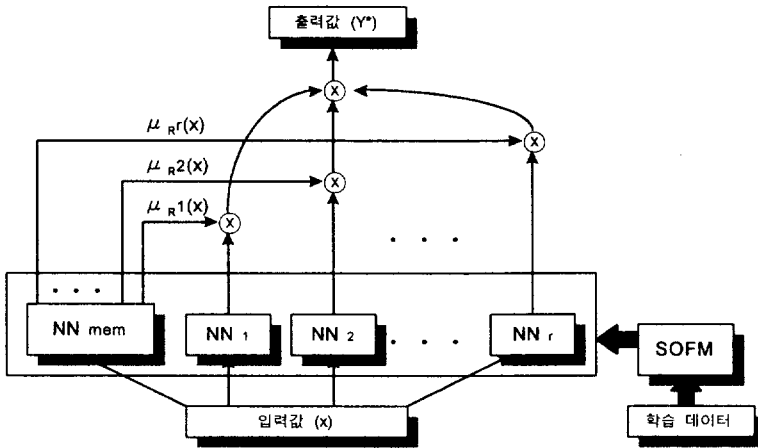
본 연구에서 제시한 퍼지근사추론 메카니즘을 예시하기 위해 사용한 실험자료는 다음의 식에 의해 산출되었다.

$$y = (1.0 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5})^2 \quad \text{if } x \leq 3.$$

$$y = (1.0 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5})^2 + (10.0 - \frac{x_3}{2.0})^2 \quad \text{if } x > 3.$$

즉 3개의 입력변수 및 1개의 출력변수로 이루어졌으며,  $x_3$ 의 임계값 3을 기준으로 출력값에 커다란 변화가 발생하는 경우이다. 이러한

식은 일반적으로 퍼지근사추론 메카니즘 실험시 이용되는 함수식으로서 주로 시뮬레이션을 통하여 자료를 생성할때에 사용된다 (Takagi



[그림 3] 제한된 퍼지근사추론 메카니즘에 의한 의사결정절차

[표 3] 학습자료 및 테스트자료

번호	학습 자료				테스트 자료			
	$x_1$	$x_2$	$x_3$	$y$	$x_1$	$x_2$	$x_3$	$y$
1	1	3	1	11.110	1	1	5	65.795
2	1	5	2	6.521	1	3	4	70.043
3	1	1	3	10.190	1	5	3	5.724
4	1	3	4	70.043	1	1	2	11.250
5	1	5	5	61.492	1	3	1	11.110
6	5	1	4	83.020	5	5	2	14.360
7	5	3	3	14.150	5	1	3	19.610
8	5	5	2	14.360	5	3	4	77.650
9	5	1	1	27.420	5	5	5	68.680
10	5	3	2	15.390	5	1	4	83.020
11	1	5	3	5.724	1	3	3	6.380
12	1	1	4	73.766	1	5	2	6.521
13	1	3	5	62.120	1	1	1	16.000
14	1	5	4	69.406	1	3	2	7.219
15	1	1	3	10.190	1	5	3	5.724
16	5	3	2	15.390	5	1	4	83.020
17	5	5	1	19.680	5	3	5	69.640
18	5	1	2	21.060	5	5	4	76.680
19	5	3	3	14.150	5	1	3	19.610
20	5	5	4	76.680	5	3	2	15.390

&Hayashi 1991).

**STEP 1**

단일 출력변수를  $y$ 로, 입력벡터를  $X = (x_1, x_2, x_3)$ 로 놓은 후, 학습자료 및 테스트 자료를 [표 3]과 같이 구하였다.  $n_t = n_c = 20$ . ([표 3]에 나타난 자료의 입력부분은 T-H 논문에서 사용된 것임)

**STEP 2**

비감독학습 인공신경망인 SOFM으로 학습자료의 입력부분을 클러스터링한 결과는 [표 4]와 같다.

[표 4] 입력부분에 대한 SOFM 학습 결과

구 분	해당하는 학습자료의 번호	
입력퍼지 집합	$A_1$	1, 2, 3, 4, 5, 11, 12, 13, 14, 15
	$A_2$	6, 7, 8, 9, 10, 16, 17, 18, 19, 20

SOFM으로 학습자료의 출력부분을 클러스터링한 결과는 [표 5]와 같다.

이상의 입력 및 출력부분에 대한 클러스터링 결과를 바탕으로 퍼지집합 관계표를 작성하고 학습자료를 해당하는 셀에 분류한 결과는 [표

6]과 같다. [표 6]에 나타난 퍼지집합 관계표의 모든 셀에 해당하는 학습자료가 존재함을 알 수 있으며 이에따라 구하고자 하는 퍼지규칙의 수는 4가 된다.

**STEP 3**

임의의 입력값  $X_j$ 에 대해 각 퍼지규칙  $R^s$ 의 적용가능성을 나타내는  $\mu_{R^s}(X_j)$ 를 구하기 위한 인공신경망 NNmem을 생성하는 단계이다. 이를 위해 [표 6]에 나타난 자료를 바탕으로 인공신경망[  $3 \times 4 \times 4 \times 4$ ]로 학습하였다. 학습된 NNmem에 테스트자료를 입력하여 얻은  $\mu_{R^s}(X_j), j = 1, \dots, 20; s = 1, 2, 3, 4$ 가 [표 7]의 둘째행부터 다섯째행까지 나타나 있다.

**STEP 4**

임의의 입력값  $X_j$ 에 대해 각 퍼지규칙  $R^s$ 의 THEN부분에 해당하는  $u_s(X_j)$ 를 구하기 위한 인공신경망 NNs,  $s = 1, 2, 3, 4$ 를 생성하는 단계이다. 이를 위해 [표 6]에 나타난 자료 및 (식 4)의 과잉학습방지 인덱스를 이용하여

[표 5] 출력부분에 대한 SOFM 학습 결과

구 분	해당하는 학습자료의 번호	
출력퍼지 집합	$B_1$	1, 2, 3, 7, 8, 9, 10, 11, 15, 16, 17, 18, 19
	$B_2$	4, 5, 6, 12, 13, 14, 20

[표 6] 퍼지집합 관계표

구 분	입 력 퍼 지 집 합		
	$A_1$	$A_2$	
출력퍼지 집합	$B_1$	$R^1 : 1, 2, 3, 11, 15$	$R^2 : 7, 8, 9, 10, 16, 17, 18, 19$
	$B_2$	$R^3 : 4, 5, 12, 13, 14$	$R^4 : 6, 20$

인공신경망[  $3 \times 4 \times 4 \times 1$  ]로 학습하였다. 학습된 NNs에 테스트자료를 입력하여 얻은  $u_s(X_j)$ ,  $j = 1, \dots, 20$ ;  $s = 1, 2, 3, 4$  가 [표 7]의 여섯째행부터 아홉째행까지 나타나 있다.

STEP 5

이상의 과정을 통해  $\mu_{R^s}(X_j)$  및  $u_s(X_j)$  가 구해지면 테스트자료에 대한 최종 추론값  $y_j^*$ 는 (식 5)를 이용하여 계산된다 ([표 8]의  $y_j^*$  참조). 본 연구에서 제시한 추론방식의 효율성을 검증하기 위해 인공신경망과 퍼지근사추론의 결합에 관련된 대표적인 연구 결과중의 하나인 Takagi & Hayashi (1991)의 방식과 비교

해 보았다. [표 3]에 나타난 자료에 대해 Takagi & Hayashi(1991)가 제시한 추론방식을 적용한 결과( $y_j^{**}$ )가 [표 8]에 나타나 있다. [표 8]서 보면 오차제곱합의 기준에 의하여 본 연구에서 제안한 방식의 추론결과가 기존의 Takagi & Hayashi(1991)가 제시한 방법보다 추론성고가 뛰어난을 알 수 있다.

5. 결론 및 향후 연구방향

본 논문에서는 비감독학습 인공신경망인 SOFM과 퍼지논리를 결합한 새로운 방식의 퍼지근사추론 메카니즘 방식을 제안하였다. 제안한 방법을 적용하여 실험한 결과 기존의 인공신

[표 7] 테스트자료에 대한  $\mu_{R^s}(X_j)$  및  $u_s(X_j)$  추론결과

번호(j)	$\mu_{R^1}(X_j)$	$\mu_{R^2}(X_j)$	$\mu_{R^3}(X_j)$	$\mu_{R^4}(X_j)$	$u_1(X_j)$	$u_2(X_j)$	$u_3(X_j)$	$u_4(X_j)$
1	0.014	0.972	0.001	0.047	9.174	14.580	70.460	81.120
2	0.043	0.970	0.001	0.020	6.951	13.960	69.660	75.780
3	0.968	0.044	0.004	0.003	6.093	14.000	73.220	73.830
4	0.981	0.001	0.045	0.001	10.487	21.200	74.160	84.120
5	0.979	0.004	0.014	0.001	9.564	18.840	74.180	78.030
6	0.001	0.001	0.970	0.020	6.860	15.160	73.900	78.480
7	0.001	0.001	0.963	0.055	8.576	15.680	73.480	83.340
8	0.001	0.028	0.055	0.960	6.444	13.340	70.700	79.440
9	0.001	0.452	0.008	0.991	5.417	13.360	65.500	75.960
10	0.001	0.041	0.044	0.968	7.952	14.020	71.580	82.830
11	0.969	0.045	0.004	0.003	7.796	14.240	73.500	76.410
12	0.980	0.017	0.006	0.002	6.652	14.540	73.900	74.220
13	0.966	0.001	0.454	0.001	10.357	27.520	74.040	84.840
14	0.980	0.013	0.007	0.002	8.706	15.120	74.080	77.160
15	0.968	0.044	0.004	0.003	6.093	14.000	73.220	73.830
16	0.001	0.041	0.044	0.968	7.952	14.020	71.580	82.830
17	0.001	0.181	0.017	0.987	5.911	13.200	63.180	78.660
18	0.001	0.044	0.042	0.968	5.742	13.460	72.060	76.830
19	0.001	0.001	0.963	0.055	8.576	15.680	73.480	83.340
20	0.001	0.001	0.970	0.019	7.848	15.580	73.860	80.880

경망이나 Takagi & Hayashi (1991) 가 제안한 퍼지근사추론 방식보다 추론결과가 크게 향상되었음을 확인하였다. 제안한 퍼지근사추론 방법의 장점을 정리하면 다음과 같다.

첫째, 미지의 자료에 대해 보다 정확한 추론이 가능하다.

둘째, SOFM방법을 입력과 출력 모두에 적용하므로써 퍼지규칙의 수를 더욱 효과적으로 결정한다.

셋째, 각 퍼지규칙을 학습함에 있어서 과잉학습 (overlearning) 방지를 위한 새로운 인덱스를 개발하였다.

넷째, 따라서, 기존의 인공신경망을 결합한 퍼지근사추론 방법과는 달리 학습속도가 크게

개선되었다.

앞으로의 연구방향은 다음과 같다.

첫째, 주어진 입력변수에 대한 보다 정확한 퍼지멤버쉽함수를 추정하는 방법의 개발이 필요하다.

둘째, 숫자가 아닌 구문화 되어있는 일반적인 퍼지규칙에 대해서도 적용가능하도록 확장되어야 한다.

셋째, SOFM이 아닌 다른 비감독학습 인공신경망 방법으로 주어진 입력 및 출력자료에 대한 보다 명확한 퍼지규칙 생성이 필요하다.

[표 8] 성과비교

번호(j)	실제값( $y_j$ )	제안된 방식 추론값( $y_j^*$ )	T-H방식에 의한 추론값( $y_j^{**}$ )
1	65.795	70.061	68.371
2	70.043	67.117	67.600
3	5.724	9.222	8.386
4	11.250	11.090	18.746
5	11.110	10.022	11.235
6	14.360	16.488	16.594
7	19.610	19.378	36.983
8	77.650	75.653	47.082
9	68.680	72.310	80.455
10	83.020	79.449	77.497
11	6.380	10.919	11.433
12	6.521	7.971	5.289
13	16.000	15.934	29.905
14	7.219	9.736	6.440
15	5.724	9.222	8.386
16	83.020	79.449	77.497
17	69.640	75.298	77.413
18	76.680	74.041	74.344
19	19.610	19.378	36.983
20	15.390	16.833	12.180
오 차 제 곱 합		169.994	2122.830

## 참 고 문 헌

- [1] Buchanan, B.G., & Shortliffe, S.H. *Rule-based expert systems*. Addison-Wesley, 1984.
- [2] Fukuda, T. & Shibata, T., "Hierarchical Intelligent Control for Robotic Motion by Using Fuzzy, Artificial Intelligence, and Neural Network," *Proc. of Int'l Joint Conf. on Neural Networks*, vol. 1(1992), pp.269-274.
- [3] Giarratano, J. & Riley, G. *Expert Systems: Principles and Programming*. PWS Publishing Company, 1994.
- [4] Graham, I. *Fuzzy logic in commercial expert systems- results and prospects*. Fuzzy Sets and Systems, Vol. 40(1991), pp.451-472.
- [5] Jang, J-S. R. "Fuzzy Controller Design without Domain Expert." *IEEE International Conference on Fuzzy Systems*, 1992, pp.289-296.
- [6] Kohonen, T., "Self-Organizing Map", *Proceedings of the IEEE*, vol. 78, No. 9(Sep, 1990).
- [7] Leung, K.S., Wong, W.S.F., & Lam, W. *Applications of a novel fuzzy expert system shell*. Expert Systems, Vol. 6(1989), pp.2-10.
- [8] McCulloch, W.S. & Pitts, W., "A logical calculus of the ideas imminent in nervous activity", *Bull. Math. Biophys*, Vol. 5(1943), pp.115-133.
- [9] Nakayama, S. Horikawa, S., Furuhashi, T., and Uchikawa, Y., "Knowledge Acquisition of Strategy and Tactics Using Fuzzy Neural Networks," *Proc. of IJCNN'92*, pp.II-751-II-756.
- [10] Negoita, C.V. *Expert systems and fuzzy systems*. The Benjamin/Cummings Publishing Company, 1985.
- [11] Pigford, D.V. & Baur, G. *Expert Systems for Business: Concepts and Applications*. Boyd & Fraser Publishing Company, 1990.
- [12] Rosenblatt, F., "The perception: a probabilistic model for information storage and organization in the brain", *Psychol. Rev.*, Vol. 65, No. 6(1958), pp.386-408.
- [13] Rumelhart, D., Hinton, S.S., & Williams, R. Learning internal representation by error propagation. in *Parallel Distributed Processing*, Vol. I, MIT Press, Cambridge, MA, 1986.
- [14] Silverman, B.G. *Expert Systems for Business*. Addison-Wesley Publishing Company, 1987.
- [15] Takagi, H. and I. Hayashi k., "NN-Driven FuzzyReasonig", *Int'l Journal of Approximate Reasoning*. Vol. 5(1991), pp.191-212.
- [16] Waterman, D.A. *A guide to expert systems*. Addison-Wesley Publishing Company, 1986.
- [17] Zadeh, L.A. *Fuzzy sets, Information and Control*, Vol. 8, No. 3(1965), pp.338-353.
- [18] Zadeh, L.A. *Outline of a new approach*

- to the analysis of complex systems and decision processes.* IEEE Transactions on Systems, Man, and Cybernetics, Vol. 3(1973), pp.20-44.
- [19] Zadeh, L.A. *The role of fuzzy logic in the management of uncertainty in expert systems.* Fuzzy Sets and Systems, Vol. 11(1983), pp.199-227.
- [20] Zaus, M. and Megent, R., "Fusion-technology and the design of evolutionary machines for neural networks", *Artificial Neural Networks, Elsevier Science Publishers B.V.(North-Holland)*, 1991, pp.1165-1168.



## 부 록

(식 5)에서 보듯이 테스트자료  $(X_j, Y_j)$ 에 대한 최종 추론값  $Y_j^*$ 는 다음과 같다 :

$$Y_j^* = \frac{\sum_{s=1}^r \mu_{R^s}(X_j) \cdot u_s(X_j)}{\sum_{s=1}^r \mu_{R^s}(X_j)}$$

따라서 테스트자료에 대한 전체 오차제곱합  $\Theta_c^T$ 는 다음과 같이 계산된다.

$$\begin{aligned} \Theta_c^T &= \sum_{j=1}^{n_c} (Y_j - Y_j^*)^2 \\ &= \sum_{j=1}^{n_c} \left\{ Y_j - \frac{\sum_{s=1}^r \mu_{R^s}(X_j) \cdot u_s(X_j)}{\sum_{s=1}^r \mu_{R^s}(X_j)} \right\}^2 \\ &= \sum_{j=1}^{n_c} \left[ \frac{\sum_{s=1}^r \mu_{R^s}(X_j) \cdot \{Y_j - u_s(X_j)\}}{\sum_{s=1}^r \mu_{R^s}(X_j)} \right]^2 \\ &= \sum_{j=1}^{n_c} \sum_{s=1}^r \left[ \frac{\mu_{R^s}(X_j) \cdot \{Y_j - u_s(X_j)\}}{\sum_{s=1}^r \mu_{R^s}(X_j)} \right]^2 \\ &\quad + \sum_{j=1}^{n_c} \sum_{t \neq u} \left[ \frac{\mu_{R^t}(X_j) \cdot \mu_{R^u}(X_j)}{\{\sum_{s=1}^r \mu_{R^s}(X_j)\}^2} \{Y_j - u_t(X_j)\} \{Y_j - u_u(X_j)\} \right] \end{aligned}$$

여기서 NNs,  $s = 1, \dots, r$  이 상호독립적으로 학습되었다는 사실을 고려하면,  $t \neq u$  일 때  $Y_j - u_t(X_j)$  와  $Y_j - u_u(X_j)$  는 상호독립이라 생각할 수 있다. 이 경우 위식의 우변의 두 번째 항의 값은 0이 된다. 따라서,

$$\begin{aligned} \Theta_c^T &\cong \sum_{j=1}^{n_c} \sum_{s=1}^r \left[ \frac{\mu_{R^s}(X_j) \cdot \{Y_j - u_s(X_j)\}}{\sum_{s=1}^r \mu_{R^s}(X_j)} \right]^2 \\ &= \sum_{s=1}^r \sum_{j=1}^{n_c} \left[ \frac{\mu_{R^s}(X_j) \cdot \{Y_j - u_s(X_j)\}}{\sum_{s=1}^r \mu_{R^s}(X_j)} \right]^2 \end{aligned}$$

이상의 과정을 바탕으로 각 NNs의 테스트자료에 대한 오차제곱합  $\Theta_c^s$ , 다시말해 테스트자료에 대한 전체 오차제곱합  $\Theta_c^T$  중 NNs에 의한 오차제곱합이 차지하는 부분은 다음과 같이 근사화될 수 있음을 알 수 있다.

$$\Theta_c^s = \sum_{j=1}^{n_c} \left[ \frac{\mu_{R^s}(X_j)}{\sum_{s=1}^r \mu_{R^s}(X_j)} \{Y_j - u_s(X_j)\} \right]^2$$