1999년도 대한전기학회 정기총회 및 추계학술대회 논문집 1999. 11. 20

컨벡스 집합을 기반으로한 클래시피케이션

박상국*. 여희주*. 김재현**
*대진대학교 전자공학과 ,**지-메이트

Convex-Set-Based Classification

Sang-Gouk Park*. Hee-Joo Yeo*. Jae-Hyun Kim**
*Dept. of Electronics Eng.,Dae-Jin Univ.. **G-mate

Abstract - 본 논문에서는 기존의 FMMCNN이나 Fuzzy ART에서 Hyperbox를 정형으로 이용한 방법보다 적응적으로 분류가 가능한 컨벡스 집합을 기반으로 한 새로운 클래시피케이션 기법을 제안하였다. 컨벡스 다면체를 적응적으로 생성하기 위하여 퍼지 뉴럴 네트웍 분류기를 구성하고, 이를 이용한 패턴 클래스들을 생성하였다. 마지막으로, FMMCNN과의 다양한 시뮬레이션을 수행하여 본 논문의 우수성을 입증하였다.

1.서 론

패턴 클래시피케이션은 많은 공학 문제들에 있어서 중요한 요소이다. 음파탐지기,레이더,지진,로보틱스,이미지 프로세싱 그리고 모든 분석 응용시스템들은 상황을 정확하게 분석하기 위한 능력을 필요로 한다. 제어, 추적, 그리고 예견 시스템들은 입력과 출력의 관계를 결정하기 위하여 분류기(classifier)를 자주 사용한다. 이러한 넓은 응용범위 때문에, 패턴 클래시피케이션[1-3]은 많은 부분에서 중요히 연구되어져 왔다. 컨벡스 다면체(convex polytope)[4]를 이용한 퍼지 클래시피케이션 방법은 기존의 FMMCNN[5-7]이나 Fuzzy ART[8]에서 제안되었던 Hyperbox를 정형으로한 클래시피케이션에 비하여 데이터 입력 순서에 많은 영향을 받았던 것을 개선한 방법으로 다음과 같이 계층적 알고리즘들을 포함한다. 입력 데이터를 나타낼 수 있는 컨벡스 다면체들을 구함으로서 초기 서브 클래스들을 형성한다. 초기 서브 클래스들을 얻기 위해 주어진 데이터를 나타내는 컨벡스 다면체를 매 입력에 대해 적응적으로 구하는 알고리즘을 만든다. 서브 클래스의 외부에 있는 입력패턴의 소속도를 계산하기 위해 비선형 함수를 사용하였다. 본 방법은 매 입력 패턴에 대해 순차적으로 수행되며, 온-라인 특성을 갖는다. 구성된 컨벡스 집합은 뉴럴 네트웍의 출력층인 클래스를 정의하기 위한 서브 클래스가 되며, 완전한 클래스 맴버쉽을 가지고 패턴 클래스 공간의 영역을 정의한다. 이러한 내용들의 각각은 본 논문에서 더욱더 상세히 설명되어 질 것이다. 따라서 본 논문은 다양한 공학 분야에서 널리 사용되고 있는 패턴 클래시피케이션의[1-3] 예로서 뉴럴 네트웍을 이용한 컨벡스 서브 클래스 집합들을 모음으로서 클래스들을 생성하는 컨벡스 집합을 기반으로한 클래시피케이션을 개발하였다.

2.컨벡스 다면체를 이용한 클래스 생성

2.1 입력 패턴이 컨벡스 서브 클래스내에 포함되는지를 결정 및 내확장성 측정

어떤 서브 클래스로부터 입력 패턴을 분리시키는 초평면의 존재성은 [4]에서 증명되었고, 이를 조사할 수 있는 알고리즘은 다음과 같이 요약되어 질 수 있다.

Separating Hyperplane Detection(SHD)

Given an input pattern x ;
FOR all vertex vectors in k-th sub-class **DO**
  Obtain expansion unit vectors e_i^k using(1) ;
END FOR
  Arbitrarily select expansion vector e_i^k as a reference vector among m expansion unit vectors ;
FOR all remaining expansion unit vectors **DO**
  Compute resultant unit vector f_i^k using(2) ;
END FOR
  Set interior = **FALSE**
  (The input pattern is initially assumed to lie outside the k-th sub-class) ;
  Set resultant unit vector pair number $t=0$;
WHILE(interior=**FALSE AND** $t \neq$ total number of resultant unit vector pairs)
  Increment t ;
  IF $f_t^k \cdot f_t^k \leq 0$ **THEN**
  Set interior =**TRUE**
  (The input pattern is in the interior side of the k-th sub-class) ;
  END IF
END WHILE

따라서, 입력패턴 x가 볼록 다면체의 안이나 밖에 놓이는지를 결정할 수 있다. 이 서브 클래스 수정 후 컨벡스 유지는 입력패턴 x가 첨가 될 경우 서브 클래스 변경의 형태를 결정하기 위한 방법을 설명한다. 이 Convexity 테스트 알고리즘은 다음과 같이 정리되어 진다[4].

Convexity Test(CT) Algorithm

Set convexity = **TRUE** ;
Construct an arbitrary sub-class C_i with all vertices except for a_i^t
(the arbitrary sub-class need not be convex) ;
Apply the SHD algorithm with a_i^t and vertices of C_i ;
IF interior = *TRUE THEN*
  Set convexity = **FALSE** ;
END IF

Convex sub class Modification (CCM) Algorithm

Construct a new sub-class with input pattern x as a new vertex and all vertices of k-th sub-class
Apply the CT algorithm with input pattern x and the k-th sub-class ;
IF convexity = **FALSE THEN**
  Discard vertex due to x from k-th sub-class ;
ELSE
  FOR all remaining vertices of k-th sub-class **DO**
    Apply the CT algorithm ;
    IF convexity = **FALSE THEN**
      Store vertex a_i^t as a vertex to be removed ;
    END IF
  END FOR
  Discard all vertices that are stored to be removed ;
END IF

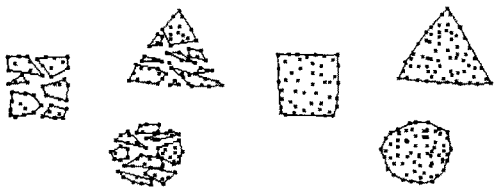
계속적으로 서브 클래스를 확장하는 것은 바람직하지 않다. 그러므로, 대부분의 클러스터링 알고리즘에서 적용되어온 것처럼, 앞에서 언급한 서브 클래스 확장 제한조건이 필요하다. 그러한 제한조건을 얻기 위해서, 서브 클래스의 크기를 얻기 위한 방법이 필요하다. 이를

위해서, 내 확장성 측정으로서 서브 클래스의 크기를 식(1)과 같이 정의한다[1].

$$\varepsilon_{intra}(A^k)=\sum_{\lambda_1=1}^{N_1}\cdots\sum_{\lambda_n=1}^{N_n} m(\lambda_1,\cdots\lambda_n)\cdot\prod_{\gamma=1}^{n} I_\gamma \qquad (1)$$

이는 컨벡스 서브 클래스의 내부에 위치한 크기 셀들의 수를 포함한다. 이것은 서브 클래스의 각 차원을 위한 작은 축 간격 I를 선택함으로 수행되어 지며, 그리고 실제와 측정된 크기사이의 에러가 최소인 내부 크기 셀들의 수를 누적함으로서 수행된다.

여기서, $I_\gamma=\max_{i\neq j}\{|a^k_{\gamma i}-a^k_{\gamma j}|/N_\gamma$ 는 γ-번째 차원을 위한 셀 길이다. N_γ 은 간격들의 수이다. 그리고 $m(\cdot)$ 은 n-D 이진화 값된 매트릭스 이다. 위 식에서, $m(\cdot)$의 값은 내부 서브 클래스 셀일경우 1이고, 그렇지 않을 경우 0이다. 크기 계산의 이 방법은 아마도 낮은 차원성의 문제 일 경우 만족된다. 그림3은 서브 클래스의 크기 파라메터 값에 따른 서브 클래스 내 확장성에 의한 변화된 결과를 보인다.



(a) (b)

그림3. (a)크기 파라메터 : 0.95 (b)크기 파라메터 : 0.84

2.2 퍼지 컨벡스 분류기 뉴럴 네트웍

뉴럴 네트웍 분야에는 서로 다른 다양한 접근 방법들이 있는데, 본 논문에서는 패턴공간에서 부분 집합들을 생성 하므로서 클래스 결정 경계들을 만드는 방법을 사용했다. 즉, 컨벡스 집합 기반의 뉴럴 네트웍은 이러한 방법을 기초로 한 뉴럴 네트웍 분류기의 예이다. 각각의 컨벡스 서브 클래스 퍼지 집합 A^k는 식(2)과 같이 순서화된 집합으로 정의 할 수 있다.

$$A^k=\{X,\ V^k, f(X,\ V^k)\} \qquad (2)$$

여기에서, X는 입력 패턴집합이고, A^k는 k-번째 컨벡스 다면체를 위한 꼭지점 집합이며, $f(X,\ V^k)$는 X와 k-번째 클래스에 소속될 소속함수를 나타낸다.따라서 j-번째 패턴 클래스 c_j는 식(3)과 같이 각 서브 클래스 A^k의 합집합으로 표현 될 수 있다.

$$C_j=\bigcup_{k\in j} A^k \qquad (3)$$

여기서 소속함수 $\mu^k(x)$는, $0\le\mu^k(x)\le1$이며, 식(4)와 같이 정의될 수 있다.[1]

$$\mu^k(x)=\frac{1}{1+\dfrac{d(x,\ x_0)^{\gamma(x_0)}}{\lambda}} \qquad (4)$$

$$d(x,x_0)=dist(x,A^k) \qquad (5)$$

$$\gamma(x_0)=\frac{N_k}{1+\dfrac{\varepsilon_{intra}(A^k)}{e_0}} \qquad (6)$$

여기서 식(5) $d(x,\ x_0)$는 소속도가 1.0인 멤버 x_0에서 입력패턴 x사이의 거리이며, λ는 $\mu^k(x)=0.5$가되는 x_0로부터의 거리를 나타내고, 식(6) $\gamma(x_0)$ ($\gamma(x_0)>0$)는 소속함수의 퍼지함(fuzziness),혹은 기울기의 급격함을 나타낸다. 또한, $dist(x, A^k)$ 는 입력 패턴 x에서 볼록 서브 클래스 A^k로 부터 최소거리를 나타내며, N_k 는 A^k의 내부안에 있거나 A^k 상에 위치한 입력패턴들의 수이고, e_0($0<e_0<1$) 는 서브 클래스 크기 측정 $\varepsilon_{intra}(A^k)$를 위한 가중치 벡터이다.k-번째 서브 클래스를 위한 입력 패턴 x의 소속도 값은 식(7)과 같이 주어진다.

$$f(k)=\begin{cases}1 & \text{서브 클래스 내부}\\ \mu^k(x) & \text{서브 클래스 외부}\end{cases} \qquad (7)$$

이제, 입력 패턴 x가 최대로 소속하는 가장 확장 할 수 있는 서브 클래스를 결정하기 위해, 동일한 클래스 특성을 갖는 서브 클래스들 사이의 가장 큰 소속도 값을 얻는 것이 필요로 하다. 이 것은 식(8)와 같다.

$$K^*=\arg\{\max_k\{f^k(x)\}\} \qquad (8)$$

여기서 K^* 는 입력패턴 x에 가장 큰 소속도를 가지는 확장할 수 있는 서브 클래스를 나타낸다. 앞에서 언급한 소속도의 할당을 사용함으로서, 서브 클래스는 다음과 같이 수행되어진다. 들어오는 입력 패턴을 위하여, SHD알고리즘은 패턴이 현재 서브 클래스들의 내부에 놓이는지를 결정하기 위해 사용되어 진다. 만일 패턴이 서브 클래스들의 하나에 속한다면, 서브 클래스 확장은 이러한 패턴일 경우 수행되지 않는다. 그렇지 않으면, 가장 확장할 수 있는 서브 클래스는 K^*를 구함으로서 얻을 수 있다. 이제, 만일 $f^{k^*}(x)$가 미리 정의된 소속도 임계값 θ 를 초과한다면, 볼록 서브 클래스 변형(CCM)알고리즘은 서브 클래스 K^*를 확장하기위해 적용되어진다. 다음, 확장된 서브 클래스의 크기는 최대 확장할 수 있는 클러스터 크기 β사이에 놓이는지를 보기 위해 검사되어진다. 마침내, 서브 클래스 K^*의 확장은 몇몇 이웃하는 다른 클래스의 서브 클래스들을 가지고 오버랩을 일으키는지를 보기 위해 검사된다, 그러한 상황을 검출 하기위해, K^*의 내부에 놓여 있는지 어떤지를 조사함으로서 쉽게 결정되어 질 수 있다. 오버랩핑의 경우 이웃하는 서브 클래스 꼭지점들의 몇몇 확장된 서브 클래스의 이러한 경우에서, SHD알고리즘은 여기서, 또한 이용되어 질 수 있다. 만일, 위조건의 하나라도 만족하지 않는다면, 확장할 수 있는 해당 서브 클래스는 주어진 입력 패턴일 경우 확장하지 않으며, 패턴 x는 새로운 점의 서브 클래스가 된다. 위 방법을 정리한 퍼지 컨벡스 서브 클래스 확장(FCSE)알고리즘은 아래와 같다.

Fuzzy Convex Sub Class Expansion (FCSE) Algorithm

```
Set all necessary initial parameter : (λ, e0, θ, β)
Given an input pattern x :
FOR all sub classes DO
    Apply SHD algorithm :
END FOR
IF interior = FALSE THEN
   Set expandable sub class number i= 0 :
   FOR all sub classes DO
       Obtain γ(x0) using (7) :
       Obtain μ^k(x) using (5) :
       IF μ^k(x)> threshold value THEN
          Increment i:
          Sort sub class indexes in decreasing
          order according to corresponding
                        μ^k(x) values :
       END IF
   END FOR
   Set expand = FALSE :
   Set expandable sub class index i=0 :
```

```
    WHILE (expand = FALSE AND i ≠ total
        number of expandable sub class)
        Increment i ;
        Expand sub class i using CCM algorithm ;
        Text following conditions for each sub class ;
        1) Determine if size of expanded
           sub class ⟨ β using (2) or (3) ;
        2) Determine if no overlap is detected
           between the expanded sub class and
           the other current sub classes using SHD
           algorithm ;
        IF the above two conditions are satisfied THEN
           Set expand = TRUE ;
           k* = expanded sub class i ;
        END IF
    END WHILE
    IF expand = FALSE THEN
        Create a new point sub class ;
    END IF
END IF
```

뉴럴 네트웍은 그림4에서 보여지는 것과 같이 퍼지 컨벡스 분류기를 처리한다. 이 세층의 뉴럴 네트웍안에서 각각의 F_B마디는 컨벡스 퍼지 집합(컨벡스 서브 클래스)을 나타낸다. 여기서, F_A와 F_B연결들은 컨벡스 집합의 꼭지점들로 구성되어진다. 이러한 모든 컨벡스 집합의 꼭지점들은 V 메트릭스에 저장 된다. V 메트릭스는 다음과 같이 식(9)처럼 나타낼 수 있다.

$$V^k=[\ V_1,\ V_1,\ \cdot\ \cdot,\ V_n] \qquad (9)$$

여기서 각각의 V_n은 컨벡스 서브 클래스들을 구성하며, 하나의 컨벡스 서브 클래스를 구성하는 꼭지점들은 식(10)과 같이 나타낼 수 있다.

$$V_1=\{\ v_{11},\ v_{12},\ v_{13},\ \cdot\ \cdot\ \cdot\ \cdot,\ v_{1n}\} \qquad (10)$$

그리고 F_B전달함수는 소속함수를 구하는 식(4)로부터 정의된 컨벡스 소속함수 이다.k-번째 F_B의 상세한 표시는 그림5에서 보여진다. 그림6는 F_B를 통과한 분류기 처리의 예를 보여준다.

F_B와 F_C,를 연결하는 마디들은 이진수 값들이고, U 메트릭스에 저장 되어 진다. F_B와 F_C,를 연결하기 위해 할당하기 위한 값들의 조건은 다음과 같다.

$$u_{kj}=\begin{cases}1 & \text{if } A^k \text{ is a convex set for class } c_j\\ 0 & \text{otherwise}\end{cases} \qquad (11)$$

여기서, A^k는 k-번째 F_B 마디이고, c_j는 j-번째 F_C 마디이다. 각각의 F_C마디는 해당 클래스를 나타낸다. F_C마디의 출력은 입력 패턴 x가 j클래스에 얼마나 해당되는지의 정도를 나타낸다. 즉 입력 패턴은 클래스를 정의 한다. F_C 마디들의 각각을 위한 전달함수는 컨벡스 퍼지 집합 값들의 퍼지 합을 수행한다. 이 식의 정의는 다음과 같다.

$$c_j=\max_{k=1}^{m} A^k\, u_{kj} \qquad (12)$$

위와 같이 이러한 뉴럴 네트웍 처리를 이용하여, 들어오는 입력 패턴에 대하여 해당하는 클래스를 정의 할 수 있다. 마지막으로 아래와 같이 본 알고리즘을 정리 하였다.

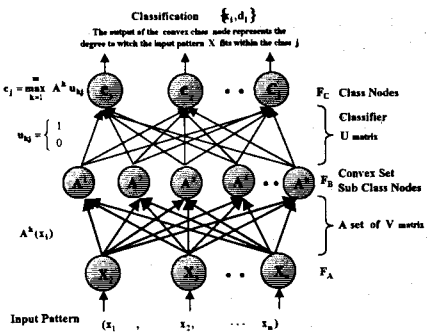


그림4. 퍼지 컨벡스 뉴럴 네트웍 구조.

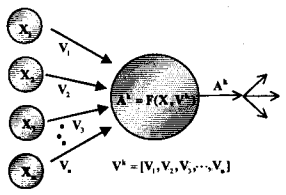


그림5. F_b마디를 위한 컨벡스 집합의 처리

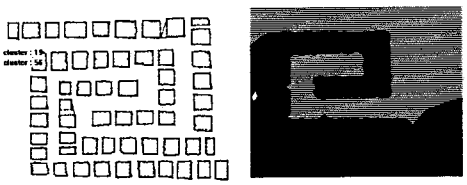


그림6. F_B를 통과한 분류기 처리

Fuzzy Convex-Set-Based Classification (FCSC) Algorithm

```
Learn to total input pattern x ;
Define class ( C_j) with learned total input pattern ;
Create an initial point sub class( A^k) for
        first training input pattern x ;
FOR remaining training input patterns x DO
    Apply FCSE algorithm ;
    Create sub classes ( A^k);
END FOR
FOR Given an total input patterns x DO
  FOR created all sub classes( A^k) DO
    Apply SHD algorithm;
    IF interior = TRUE THEN
      FOR defined classes ( C_j) DO
        IF sub class ( A^k) is a convex set
                    for class( C_j) THEN
          U_kj = TRUE;
          Set C_j = max A^k;
        END IF
      END FOR
    END IF
  END FOR
END FOR
```

3. 실험 및 결과

제안된 방법의 효율성을 입증하기 위하여 몇 가지 실험 결과를 제시하고자 한다. 실험에서 사용된 두 알고리즘은 FCSC와 FMMCNN[4]를 알고리즘으로한 시뮬레이션 결과이다. 모든 입력패턴 벡터들은 1600개의 데이터 집합으로 그림7과 같은 입력패턴들을 사용하였으며,

이 데이터 집합은 두 클래스로 나누어 학습을 시킨 다음 각각의 클래스에 해당하는 모든 입력패턴에 클래스 레이블을 주었다.(그림8).이러한 데이터 집합 중에서 순서에 상관없이 랜덤하게 10%,15%,30%,50%의 트레이닝 데이터 집합을 추출하여 분류기를 구성해 클래시피케이션한 결과를 얻었다. 아래 그림9는 10%의 트레이닝 데이터 집합의 예를 보여준다.

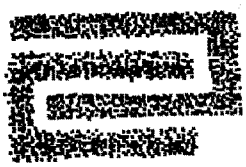


그림7. 모든 데이터 집합

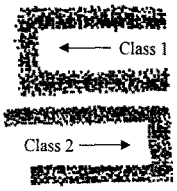


그림8. 두 클래스 데이터 집합

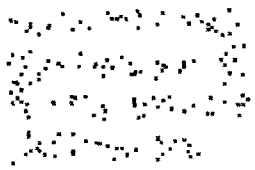


그림9. 트레이닝 데이터 집합:160(10%)

제일먼저 트레이닝 데이터 집합 160(10%)을 두 알고리즘 각각의 서브 클래스 크기 파라메터를 이용하여 적용한 결과 생성된 서브 클래스들의 수에서 FCSC는 26개,FMMCNN는 32개를 생성하였다(그림10,11). 여기에서 생성된 서브 클래스들의 수는 적으면 적을수록 정확한 클래시피케이션을 하기가 어려우며 또한 트레이닝 데이터 집합은 많으면 많을수록 좋은 결과를 얻을 수 있을 것이다. 두 결과에서 FCSC에서는99.68%라는 정확도를 얻을 수 있었으며, FMMCNN에서는 91.37%의 정확도를 얻었다. 좀더 명확한 실험 결과를 얻기 위하여 트레이닝 데이터 집합 30%를 사용하였다. 그림12,13에서와 같이 FCSC는 서브 클래스 개수 37개에 99.93%라는 정확도를 얻었으며 FMMCNN은 서브 클래스 개수 39개 임에도 불구하고 94.5%이라는 정확도를 얻었다. 트레이닝 데이터집합을 더욱더 증가시킨 그림14,15는 50%를 사용했을때의 결과를 보여준다. 마지막으로 그림16,17은 생성된 서브 클래스 수에 따른 미스클래시피케이션 에러수를 보여 주며, 이를 그래프로 나타내었다. 결과에서 FCSC 알고리즘이 FMMCNN에 비하여 더욱더 좋은 결과를 얻을 수 있다는 것을 보여주고 있다.

4. 결론 및 연구과제

본 논문에서는 기존의 FMMCNN이나 Fuzzy ART에서 Hyperbox를 정형으로 이용한 방법보다 적응적으로 분류가 가능한 컨벡스 집합을 기반으로 한 새로운 클래시피케이션 기법을 제안하였다. 컨벡스 다면체를 적응적으로 생성하기 위하여 퍼지 뉴럴 네트워 분류기를 구성하고, 이를 이용한 패턴 클래스들을 생성하였다. 본 논문에서 적용한 FCSC는 FMMCNN에서의 하이퍼 박스보다 더욱더 효과적인 컨벡스 다면체를 사용함으로서 제시한 방법의 타당성과 우수성을 시뮬레이션을 통하여 입증하였다. 추후 연구과제로서는 컨벡스 다면체 크기를 적확히 얻을 수 있는 방법이 개발되어져야 하며, 적절한 파라메터 값들의 선택에 관한 분석이 이루어져야 할 것이다.

[참 고 문 헌]

[1] P.Devijver and J.Kittler, "Pattern Recognition :A Statistical Approach.Englewood Cliffs,"*NJ:Pretice-Hall*,1982.

[2] K.Fukunaga,"Introduction to Staticstical Pattern Recognition," *New York: Academic Press*,1972.

[3] S. Abe and M. Lan, "A Method for Fuzzy Rules Extraction Directly from Numerical Data and its Application to Pattern Classification,"*IEEE Trans. Fuzzy Syst.*,vol.3,pp.18-28, Feb.1995.

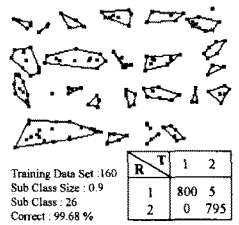
[4] I.H. Suh, J. H. Kim, and F. C. H.Rhee,"Convex-Set-Based Fuzzy Clustering," *IEEE Trans. Fuzzy System*, vol,7,pp.271-285,June.1999.

[5] P. Simpson,"Fuzzy Min-Max Neural Networks Part1:Classification," *IEEE Trans. Neural Network," vol.*3, pp.766-786, Sept. 1992.

[6] P. Simpson,"Fuzzy Min-Max Neural Networks Part 2 : Clustering,"*IEEE Trans. Fuzzy Systems* vol. 1, pp.32-45,Feb.1993.

[7] P. Simpson,"Fuzzy Min-Max Neural Networks,"in Proc. Int. *Joint Conf. Neural Network*, pp.1658-1669, Nov.18-21, 1991.

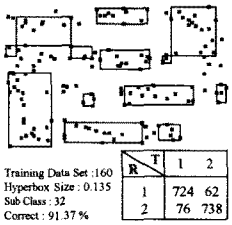
[8] G.Carpenter,S.Grossberg,andD.Rosen,"Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by Adaptive Resonance System," *Neural Network*, vol.4,pp.759-771,1991.



Training Data Set :160
Sub Class Size : 0.9
Sub Class : 26
Correct : 99.68 %

R \ T	1	2
1	800	5
2	0	795

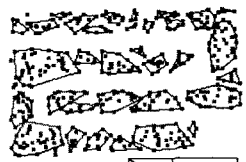
그림10. FCSC : 10%



Training Data Set :160
Hyperbox Size : 0.135
Sub Class : 32
Correct : 91.37 %

R \ T	1	2
1	724	62
2	76	738

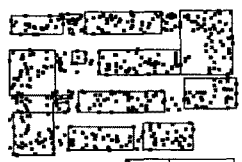
그림11.FMMCNN : 10%



Training Data Set :480
Sub Class Size : 0.945
Sub Class :37
Correct : 99.93 %

R \ T	1	2
1	800	1
2	0	799

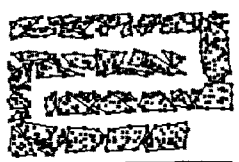
그림12. FCSC : 30%



Training Data Set :480
Hyperbox Size : 0.155
Sub Class : 39
Correct : 94.5 %

R \ T	1	2
1	765	53
2	35	747

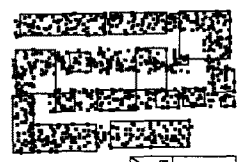
그림13. FMMCNN : 30%



Training Data Set :800
Sub Class Size : 0.965
Sub Class :47
Correct : 100 %

R \ T	1	2
1	800	0
2	0	800

그림14. FCSC : 50%



Training Data Set :800
Hyperbox Size : 0.155
Sub Class : 53
Correct : 91.43 %

R \ T	1	2
1	681	18
2	119	782

그림15. FMMCNN : 50%

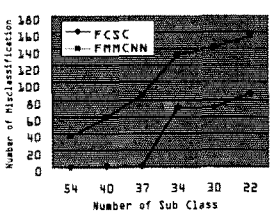


그림16. 트레이닝 데이터 : 30%

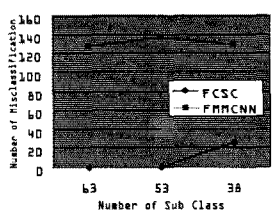


그림17. 트레이닝 데이터 : 50%