공학석사학위논문

패턴 분류를 위한 가변길이 서브패턴 집합의 진화적 최적화

Evolutionary Optimization of a Collection of Variable-Length Subpatterns for Pattern Classification

2008년 8월

서울대학교 대학원
컴퓨터공학부
김 주 경

패턴 분류를 위한 가변길이 서브패턴 집합의 진화적 최적화

Evolutionary Optimization of a Collection of Variable-Length Subpatterns for Pattern Classification

지도교수 장 병 탁

이 논문을 공학석사학위논문으로 제출함

2008년 5월

서울대학교 대학원

컴퓨터공학부

김 주 경

김주경의 석사학위논문을 인준함

2008년 6월

위 원 장 문 병 로 (인)

부 위 원 장 장 병 탁 (인)

위 원 **Robert Ian McKay** (인)

초 록

규칙 기반 분류기(rule-based classifier)의 학습 결과로 생성되는 규칙들은 쉽게 파악할 수 있고 분류와 관련성이 적은 규칙들은 직접 제어할 수 있다는 장점이 있다. 그러나 기존 기법들은 주로 지역 최적화된 해법을 제시하며, 결정 경계(decision boundary)가 입력 공간 좌표축에 대해 수직 또는 수평인 초입방체(hypercube)들의 집합 형태로만 표현된다는 단점이 있다.

본 논문에서는 위의 문제점들을 보완한 서브패턴 집합 기반의 분류 모델을 제시한다. 여기에서 서브패턴은 각 학습패턴에서 추출된 임의의 부분집합이다. 서브패턴들은 임의로 생성된 후, 적응도에 따라 유지 또는 교체되며 이 과정에서 다양한 서브패턴 조합이 시도됨으로서 지역 최적해보다 좋은 결과를 낼 가능성이 있다. 또한 제안 방법은 초입방체 형태 외에 유클리드 거리를 기준으로 한 초구체(hypersphere) 결정 경계를 허용함으로써 보다 다양한 형태의 결정경계를 표현할 수 있다. UCI 데이터를 사용한 분류 실험에서 본 제안 방법은 기존 분류기들과 비등한 정확도를 보여주었으며, 특히 높은 복잡도를 가지는 문제에서 높은 성능을 나타내었다.

주요어 : 규칙 기반 분류기, 결정 경계, 서브패턴, 랜덤 샘플링

학 번 : 2006-21175

목 차

그림·표 목차

Ⅰ. 서론

규칙 기반 분류기(rule-based classifier)는 학습의 결과로 생성된 규칙들을 분류에 사용한다. 이 규칙들은 표 1처럼 사람이 보고 이해하기 쉬운 형태로 구성되기 때문에 학습 결과물의 해석이 쉽고 잘못된 규칙이 학습되었을 경우 임의로 삭제할 수 있으며 따로 사전지식을 가지고 있을 경우 규칙으로 삽입함으로서 쉽게 적용을 할 수 있다. 또한 입력패턴의 모든 속성들을 사용할 필요 없이 입력 공간상의 위치에 따라 분류에 관련 있을 가능성이 높은 속성들만 추려서 사용할 수 있으므로 패턴 분류 시 관련 없는 속성이 노이즈로 작용할 소지를 줄일 수 있다. 다른 종류의 분류기들은 특정 속성에 가중치를 주거나 앙상블 학습 등의 방법을 이용하여 관련 없는 속성의 효과를 줄이고자 하지만 특정 속성이 입력 공간 내의 특정 영역에서는 의미가 있고 다른 영역에서는 의미가 없을 수 있기 때문에 그러한 일괄적인 처리 방법은 그 효과가 제한적일 수 있다.

표 1. 학습 규칙들의 예

(날씨=맑음, 온도=높음, 습도=높음, 바람=약함) → 테니스 안침
(날씨=맑음, 온도=낮음, 습도=보통) → 테니스 침
(날씨=흐림, 습도=높음, 바람=강함) → 테니스 안침

반면 기존 규칙 기반 분류기들은 각각의 지역 최적화 알고리즘을 이용하여 규칙 집합을 찾아내므로 분류를 위한 최적의 규칙들을 찾지 못할 수 있다. 또한 각 속성이 실수 값을 가지는 경우 그 값이 특정 범위 안에 있는지의 여부를 다른 속성과는 별개로 판단하고 모든 속성들이 각자의 주어진 조건을 만족할 때 해당 규칙이 유효한 것으로 판정하므로 결정 경계(decision boundary)가 그림 1처럼 수직 또는 수평인 초입방체(hypercube)들의 집합의 형태로만 이루어지게 되어 표현 가능한 결정 경

계가 제한적이게 된다.

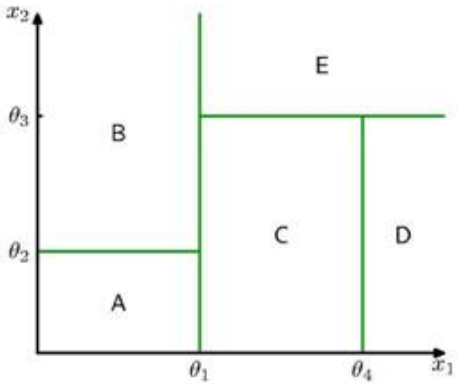

그림 1. 초입방체들의 집합 형태의 결정 경계

본 논문에서는 규칙 선택 시 지역 최적해에서 벗어날 수 있고 다양한 형태의 결정 경계를 가짐으로서 더 높은 분류 정확도를 보일 수 있는 분류 모델을 제시한다. 우선 각 학습패턴을 기반으로 임의의 속성들을 포함하는 부분집합인 서브패턴들을 생성한다. 서브패턴은 규칙 기반 분류기에서의 규칙에 대응된다. 서브패턴이 생성된 후 각 서브패턴마다 학습패턴의 분류에 대한 적응도를 계산하여 적응도가 낮은 서브패턴은 다시 임의로 생성된 서브패턴으로 교체한다. 이 과정을 반복 시 적응도가 높은, 즉 패턴분류에 뛰어난 서브패턴들만 남게 되고 이들을 가지고 분류를 수행하게 된다. 이 과정은 임의의 후보군 생성, 적응도 계산, 선택, 후보군 재생성의 과정들을 가지므로 지역 최적해를 벗어난 더 높은 정확도를 가지는 규칙 집합을 찾아낼 가능성이 있다.

또한 제시하는 모델은 기존의 초입방체 형태의 결정 경계 뿐 아니라 입력패턴과 규칙의 대응되는 속성 값들 간의 유클리드 거리를 계산하고 그 값에 따라 규칙이 만족되는지 여부를 판정하는 방식으로 초구체

(hypersphere) 형태의 결정 경계도 가질 수 있다. 분류기의 결정 경계를 초입방체 형태와 초구체 형태를 혼용하여 보다 다양하게 구성할 수 있으므로 보다 적절한 결정 경계의 구성에 따른 정확도 향상 효과를 볼 수도 있다.

이와 유사한 방법으로는 패턴의 속성들이 이진 혹은 범주 값을 가지는 경우에 대해 진화 연산 개념을 도입하여 고정 길이의 규칙 집합 [1] [2] [3] 또는 다양한 길이의 규칙집합을 찾고자 하는 모델들이 제시되어 있다 [4].

본 논문의 구성은 다음과 같다. 2장에서는 규칙 기반 분류기 관련 연구 및 배경에 대한 소개를 한다. 3장에서는 제시하는 모델의 학습 및 분류에 대한 설명을 한다. 4장에서는 UCI machine learning repository [6]에서 속성의 개수가 30개 이상, 패턴의 수가 200개 이상인 패턴들에 대한 분류 성능을 실험해본다. 5장은 결론으로서 본 논문의 연구 내용과 결과를 요약하고 후속 연구의 방향에 대해 언급하며 끝을 맺는다.

Ⅱ. 배경 연구

2.1. 규칙(Rule)

일반적인 분류 모델에서 규칙은 **전제(antecedent) → 결론(consequent)**의 형태로 구성된다. 전제는 패턴의 속성과 그 속성이 가져야 할 값의 쌍들의 집합으로 구성되고 결론은 전제의 조건이 만족 시 나타낼 클래스를 나타낸다.

예를 들어 **(날씨=맑음, 습도=높음, 바람=약함) → 테니스 안침** 이라는 규칙이 있을 때 → 의 왼쪽은 전제, 오른쪽은 결론이 되고 전제의 날씨, 습도, 바람은 패턴의 속성들, 맑음, 높음, 약함은 각 속성이 가져야할 값이 된다. 이러한 규칙이 있을 때 입력패턴이 **(날씨=맑음, 온도=높음, 습도=높음, 바람=약함)** 라면 규칙에도 명시되어 있는 **날씨**, **습도**, **바람** 속성의 규칙과 입력패턴에서의 값들이 모두 일치하므로 이 규칙은 유효한 규칙으로 인정되고 **테니스 안침** 으로 클래스를 결정하게 된다. 만약 하나의 속성이라도 값이 일치하지 않았다면 그 규칙은 입력패턴의 클래스 판정에 영향을 주지 않게 된다.

패턴의 속성들이 이진 값이나 범주 값을 갖는 경우에는 위와 같은 형태로 속성 값의 일치 여부를 판정하지만 속성들이 실수 값을 가질 수 있는 경우엔 해당 속성의 값이 어떤 범위에 있는지를 기준으로 규칙의 유효여부를 따지게 된다. 예를 들어 **(나이<25, 65<무게<75, 시력>1.0) → 1급** 의 형태로 규칙을 나타낼 수 있다. 이 경우 입력으로 들어온 패턴이 **(나이=23, 키=180, 무게=68, 시력=1.2)** 라면 클래스를 1급으로 결정하게 된다.

2.2. 규칙 기반 분류(Rule-Based Classification)

규칙 기반 분류기들은 2.1에서 설명한 규칙들의 집합을 이용하여 분류를 수행하게 된다. 규칙 집합을 생성해내는 알고리즘 중 가장 기본적인 형태로 Sequential Covering 알고리즘이 있다. 이 알고리즘은 전체 학습패턴과 비어있는 규칙 집합으로부터 시작하여 최대한 많은 학습패턴을 만족시키는 규칙들을 만들고 규칙 집합에 넣고 만족된 학습패턴은 제거한 후 나머지 학습패턴들에 대해서 다시 규칙을 만들어 넣는 것을 반복하여 규칙 집합을 구성한다. Sequential Covering 알고리즘에는 실제 구현 방법에 따라 AQ [7], CN2 [8], RIPPER [9] 등이 있다.

2.3. 귀납 논리 프로그래밍(Inductive Logic Programming)

2.2의 규칙 기반 분류기에서 규칙의 전제들의 속성들은 모두 특정 값 또는 범위가 지정되어 있고 이들 규칙은 명제 규칙(propositional rule)들로 나타낼 수 있다. 이와 달리 귀납 논리 프로그래밍에서는 각 속성이 특정 값 또는 범위로 제한되지 않고 변수를 도입해서 속성 값들의 관계를 추가적으로 표현할 수 있는 1차 혼 규칙(first-order Horn rule)을 학습하여 사용할 수 있다. 대표적인 귀납 논리 프로그래밍 모델로 FOIL (First-Order Inductive Learner) 이 있다 [10]. FOIL은 기본적으로 Sequential Covering 알고리즘과 유사한 형태를 가지고 있는데 각 속성을 추가할 때 속성을 변수로 하여 가능한 경우들을 나열해 보아 성능이 좋은 것을 채택하여 규칙에 추가하는 방향으로 진행한다.

귀납 논리 프로그래밍은 관계학습(Relational Learning)이 가능하다는 장점이 있지만 가능한 규칙의 표현형이 많으므로 많은 속성을 가지는 대규모 데이터에 대해서는 학습 수행속도가 문제가 될 수 있다.

2.4. 연관 규칙 기반 분류(Association Rule-based Classification)

데이터 마이닝 분야에서 주어진 데이터에서 빈번하게 출현하는 연관 규칙을 효율적으로 찾는 방법이 많이 연구되어 왔다. 연관 규칙을 발견하는 알고리즘을 이용해서 패턴 분류에 사용할 수 있는 분류 규칙 집합을 찾는 방법들로 CBA (Classification Based Associations) [11], CMAR (Classification Based on Multiple Association Rules) [12], CPAR (Classification Based on Predictive Association Rules) [13], MCAR (Multi-class Classification based on Association Rule) [14] 등이 제시되어 있다.

CBA는 연관 규칙 집합을 너비 우선 탐색(Breadth First Search)으로 검색하는 가장 기본적인 알고리즘인 Apriori [15] 의 방법론을 이용하여 특정 support, confidence가 넘는 규칙들을 이용해서 분류를 수행한다.

CMAR는 너비 우선 탐색의 문제점을 해결하고자 깊이 우선 탐색(Depth First Search)으로 연관 규칙을 검색하는 FP-tree [16] 의 방법론을 이용하여 분류 규칙들을 찾는다.

CPAR는 FOIL과 비슷한 방식으로 규칙을 찾는데 FOIL과의 차이점으로는 일단 만족된 학습패턴을 배제하도 다음 단계를 진행하는 게 아니라 학습패턴의 가중치를 조정해 가는 방법으로 규칙들을 찾아간다. 그리고

조건을 만족하는 모든 규칙들을 사용하지 않고 분류에 성능이 좋을 것으로 예상되는 일부 상위 규칙들만 생성해서 사용하게 되므로 CMAR보다 효율적이다.

MCAR은 연관 규칙 생성 후 규칙에 지지도(support), 신뢰도(confidence), 규칙의 길이 등의 다양한 기준을 이용하여 랭킹을 측정 후 높은 랭킹을 가진 규칙들을 분류에 사용하는 방법으로서 분류 성능을 향상 시킨다.

2.5. 결정 트리(Decision Tree)

결정 트리는 주어진 학습패턴으로부터 패턴 분류에 가장 영향이 큰 속성을 선택하여 노드를 만들고 그 속성이 가질 수 있는 값에 따라 트리의 줄기를 뻗친 후 다시 또 속성을 선택해서 노드를 추가해 나가는 방식으로 트리를 구성하는 방식으로 학습이 이루어진다. 이 때 루트 노드부터 단말 노드까지의 경로를 하나의 규칙으로 볼 수 있다. 속성을 선택하는 방법에는 ID3 [17], C4.5 [18], CART [19] 등의 결정 트리를 구성하는 알고리즘에 따라 information gain, gain ratio, gini index 등의 다양한 종류가 있다. 그림 2는 학습이 완료된 결정 트리의 예이다.

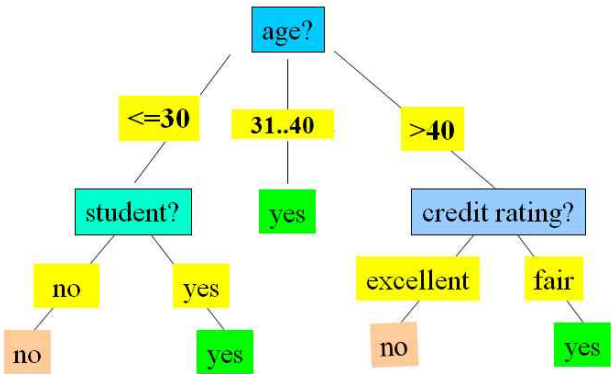

그림 2. 결정 트리의 예 [20]

입력패턴이 주어지면 루트 노드로부터 시작하여 부합하는 속성 값에 따라 트리를 검색 후 말단 노드에 도착 시 지정되어 있는 클래스로 분류를 하게 된다.

결정 트리는 규칙들의 집합을 트리 형태로 묶어서 나타내므로 사람이 보고 이해하기 쉽고 효율적으로 구현이 가능하다. 하지만 트리를 풀어서 규칙들의 집합으로 볼 경우 상위 노드의 속성들은 무조건 규칙으로 포함되게 되는데 이들이 입력 공간상의 특정 영역의 분류를 수행 시엔 필요 없는 속성일 수 있으므로 노이즈 역할을 할 수 있다는 단점이 있다.

Ⅲ. 가변길이 서브패턴 집합의 최적화

3.1. 학습 및 분류 알고리즘

본 논문에서 제시하는 가변길이 서브패턴 집합(CVLS) 모델의 학습 알고리즘은 다음과 같다.

입력: n개의 원소를 가지는 학습패턴 집합 $X = \{x_1, x_2, ..., x_n\}$
출력: d개의 원소를 가지는 서브패턴 집합 $S = \{s_1, s_2, ..., s_d\}$

1. 학습패턴들을 정형화(Standardization)하고 서브패턴 길이 선택의 확률 분포 D와 결정 경계 선택의 확률 분포 B를 균등 분포(uniform distribution) 로 초기화

2. 학습패턴의 임의의 속성들을 포함하는 부분집합인 서브패턴들을 d개 생성하여 서브패턴 집합 S를 구성. 이 때 서브패턴의 길이와 결정 경계는 각각 D와 B의 확률 분포에 따라 결정됨

3. 서브패턴 집합을 이용하여 학습패턴들을 분류함. 올바르게 분류된 학습패턴들은 Xc, 틀린 학습패턴들은 Xw 에 넣음. 모든 학습패턴을 바르게 분류 시 종료

4. 각 서브패턴 원소의 적응도를 계산

5. D와 B를 적응도 높은 서브패턴 $|S| \times |Xc|/|X|$개의 길이 분포와 결정 경계 분포로 갱신

6. 적응도 낮은 하위 서브패턴 $|S| \times (1 - |Xc|/|X|)$ 개를 학습패턴으로부터 새로 생성한 서브패턴으로 교체. 이 때 생성되는 서브패턴의 길이의 확률과 결정 경계 선택 확률은 각각 D와 B를 따른다.

7. 3단계로 돌아감. 지정된 횟수 이상 loop이 반복되었으면 종료

그림 3. 학습 알고리즘

3.2. 서브패턴의 생성

하나의 서브패턴은 특정 학습패턴의 부분집합으로서 분류기의 규칙의 후보가 된다. 서브패턴은 학습패턴 상의 특정 속성의 **인덱스=속성값**들과 클래스로 구성된다. 예를 들어 주어진 학습패턴에 대해 생성될 수 있는 서브패턴들의 예는 다음과 같다.

2개의 5차원 학습패턴 (0,1,0,0,1,true), (1,1,1,0,0,false) 존재할 경우
(f_1=0, f_3=0, f_5=1, *class*=true)
(f_2=1, f_4=0, *class*=true)
(f_2=1, f_3=1, f_4=0, *class*=false)
(f_1=1, f_2=1, f_4=0, f_5=0, *class*=false)

그림 4. 서브패턴 생성 예

서브패턴의 길이는 확률분포 D에 의해 결정된다. 처음에는 D의 분포가 균등하므로 특정 길이가 선택될 확률은 $\frac{1}{\text{패턴의 차원}}$이 되고 반복문을 수행해가면서 확률분포가 변하게 된다. 마찬가지로 서브패턴의 결정경계의 형태는 확률분포 B에 의해 결정된다. 처음에는 결정 경계가 초입방체가 될 확률과 초구체가 될 확률이 $\frac{1}{2}$로 동일하지만 반복문을 수행해가면서 확률분포가 변하게 된다.

학습패턴에서 어떤 속성이 서브패턴에 포함 될 것인지를 결정할 때 우선 학습패턴의 특정 속성 값이 없는 경우, 즉 missing value가 있는 경우 그 속성은 서브패턴에 포함시키지 않는다. 많은 분류기들이 이러한 경우 거리계산을 적절히 할 수 없는 등 처리방법이 애매해지는 점 때문에 해당 속성 값을 적절한 값으로 채운 후 분류를 수행하게 되는데 이렇게 채워놓은 값 때문에 학습 모델 상에서 거리의 오차가 생기는 등의 문제가

생길 수 있다. 하지만 서브패턴을 사용하는 모델에서는 해당 속성은 그냥 포함을 시키지 않고 다른 속성들만을 이용하면 되므로 없는 속성 값을 채움으로서 발생할 수 있는 오차의 영향을 받지 않으므로 속성 값이 없는 경우가 많은 학습패턴이 주어졌을 때 다른 분류기들 보다 유리하게 분류를 수행할 수 있다.

한편, 서브패턴을 생성 시 학습패턴의 특성에 따라 속성의 임의적인 선택에 추가적으로 적절한 휴리스틱을 이용하면 보다 효율적, 효과적으로 서브패턴을 구할 수 있다. 예를 들어 사람의 얼굴 데이터가 학습패턴으로 들어오는 경우 얼굴에서 패턴 분류에 중요한 역할을 하는 눈, 코, 입 등에 해당하는 영역들을 서브패턴에 포함하게 하는 경우 보다 효율적으로 적절한 서브패턴 집합을 구성할 수도 있다. 또는 이미지 상에서 세그먼트에 해당하는 영역을 서브패턴에 포함하게 하는 것도 적절한 서브패턴을 찾는데 힌트가 될 수 있다.

3.3. 서브패턴의 결정 경계 표현

제안 모델에서는 서브패턴 $s_i = \left(s_{i_1} = a,\ s_{i_2} = b, \ldots,\ s_{i_d} = c\right) \rightarrow class s_i$ 가 있을 때 입력패턴 p에 대해 식 (1)과 같이 서브패턴의 각 속성별로 따로 조건 만족 여부를 확인하여 모든 속성들이 만족 시 해당 서브패턴을 유효한 것으로 판정하는 방식으로 기존 규칙 기반 분류기 같이 초입방체 형태의 결정 경계를 표현 할 수 있다.

$$\forall_j \left(s_{i_j} - p_{i_j}\right) < \frac{1}{r} \rightarrow class s_i \qquad (1)$$

이 외에 서브패턴의 속성 값들과 입력패턴의 대응되는 속성 값들 사이

의 유클리드 거리를 계산하여 그 거리에 따라 유효한 지를 판정하게 함으로서 초구체 형태의 결정 경계를 표현할 수 있다. 이 경우 서브패턴의 길이, 즉 서브패턴에 포함된 속성들의 개수를 고려하지 않고 유클리드 거리를 계산하면 속성들의 수가 적은 서브패턴들에 대해 유효 거리가 편향되게 짧게 되고 유효한 서브패턴으로 만들 가능성을 높일 수 있으므로 유효 거리에 $\sqrt{\text{서브패턴의 속성개수}}$ 를 곱함으로서 속성 개수에 대한 편향성을 완화시킨 값을 기준으로 서브패턴의 유효성을 판단한다. 즉, 식 (2) 같이 조건을 나타낼 수 있고 거리가 지정된 값 미만인 경우 유효한 것으로 판정한다.

$$\sqrt{\sum_{j=1}^{d}\left(s_{i_j}-p_{i_j}\right)^2} < \frac{\sqrt{s_i\text{의 속성 개수}}}{r} \rightarrow class_i \qquad (2)$$

위의 두 가지 식에서 r은 기준 거리를 결정하는 매개변수로서 일반적으로 크기가 클수록 조건을 만족하는 서브패턴의 수가 줄어들게 된다.

한편, 특정 속성에서의 거리 값이 일방적으로 매우 클 경우 그 속성 값에 의해서만 일방적으로 서브패턴의 만족여부가 판정되는 걸 방지하기 위해 알고리즘 상에서 모든 패턴의 속성들을 우선 정형화 한 후 분류를 수행하게 된다.

3.4. 분류 알고리즘

서브패턴 집합과 입력패턴이 주어질 때 입력패턴의 분류를 수행하는 알고리즘은 다음과 같다.

입력: 서브패턴 집합 $S=\{s_1, s_2, ..., s_d\}$, m차원 입력패턴 $p=\{p_1, p_2, ..., p_m\}$
출력: 입력패턴의 클래스

1. 입력패턴에 대해 서브패턴에 정해진 결정 경계에 따라 (1) 또는 (2) 식을 만족하는 모든 서브패턴들을 추출

2. 추출된 서브패턴들의 클래스 중 가장 많은 클래스를 입력패턴의 클래스로 정함(majority voting)

그림 5. 패턴 분류 알고리즘

학습 알고리즘의 3단계에서 현재 서브패턴 집합을 이용해서 학습패턴을 분류하는 것은 현재 서브패턴 집합의 분류 정확도를 평가하기 위해서이다. 대부분의 분류 알고리즘들은 기본적으로 학습패턴의 분포와 테스트패턴들의 분포가 비슷할 것이라는 가정 하에 학습을 수행한다. 그러므로 학습패턴들을 입력패턴으로 넣고 분류 수행시의 분류 정확도와 별도의 패턴을 입력 시의 분류 정확도 간의 상관관계가 클 것으로 가정할 수 있고 이를 분류 성능의 기준으로 사용할 수 있다.

단 학습패턴에 대한 분류 정확도를 극대화 하는 것에만 초점을 맞추어 학습을 진행할 경우 학습패턴을 외워버리는 방식으로 학습이 진행되어서 학습패턴에 대한 정확도는 올라가도 별도의 패턴에 대한 정확도는 오르지 않거나 오히려 떨어뜨릴 수도 있다. 이를 과적합(overfitting)이라 하는데 이를 완화하려면 모델을 학습시킬 때 사용하는 학습패턴들과 모델의 정확도를 측정할 때 사용하는 검증패턴들을 별도로 두어서 학습을 진행하는 방법이 있다.

제시하는 학습 알고리즘에서는 3단계에서 학습패턴들의 분류 정확도를 측정할 때 특정 학습패턴을 분류 시 그 학습패턴을 원판으로 하여 만들어진 서브패턴의 경우에는 분류에 사용하지 않도록 한다. 이 경우 각 서브패턴들은 직접적인 관련이 없는 다른 학습패턴들에 대해서만 분류에

참여하게 되므로 모든 다른 학습패턴들을 검증패턴으로 사용하는 효과가 있게 되어 과적합을 완화하는 효과를 볼 수 있다.

3.5. 서브패턴의 적응도 계산

식 (3)은 각 서브패턴의 적응도를 구하는 공식이다.

$$fit(s_i) = \alpha \frac{mr(s_i)}{mw(s_i) + mr(s_i)} + (1-\alpha) \frac{cr(s_i)}{cw(s_i) + cr(s_i)} \quad (3)$$

학습 알고리즘 3단계에서 잘못 분류된 학습패턴 중 서브패턴 s_i가 조건을 만족시키고 클래스가 다른 패턴 개수는 *mw*, 클래스가 같은 패턴 개수는 *mr*이다. 마찬가지로 바르게 분류된 학습패턴 중 서브패턴 s_i가 조건을 만족시키고 클래스가 다른 패턴 개수는 *cw*, 같은 패턴 개수는 *cr*이다.

여기에서 $\frac{mr(s_i)}{mw(s_i) + mr(s_i)}$는 잘못 분류된 학습패턴에 대한 정확도, $\frac{cr(s_i)}{cw(s_i) + cr(s_i)}$는 바르게 분류된 학습패턴에 대한 정확도를 나타내고 α는 두 정확도의 적응도에 대한 기여도 비율을 지정하는 상수로서 0 이상 1 이하의 값을 갖게 된다. 정확도 부분을 분리해서 계산하는 이유는 학습 알고리즘의 반복문이 돌면서 적응도에 따라 서브패턴들이 교체될 때 그 다음 반복문에서는 현재 시점에서 분류를 제대로 못한 학습패턴들을 잘 분류할 수 있는 서브패턴의 적응도를 더 높게 하기 위해서이다. 이는 부스팅(boosting)에서 반복문이 돌면서 전 단계에서 잘못 분류한 학습패턴들에 대해 그 다음 단계에서 가중치를 높여 주는 방식으로 반복문을 진행함에 따라 최대한 많은 학습패턴이 바르게 분류될 수 있도록 하는 것

과 비슷하다 [21]. 그러지 않고 모든 학습패턴에 대해 동일한 비중으로 적응도를 계산하면 특정 클래스의 밀도가 높은 영역의 학습패턴들에 부합하는 서브패턴들의 적응도만 높게 되어 밀도가 낮은 영역에 분포한 학습패턴에 부합하는 서브패턴들은 교체가 되어 버림으로 해당 영역의 패턴에 대한 분류 정확도가 낮아지므로 전체적인 분류 정확도의 향상이 미진해질 수 있다. 제안 알고리즘처럼 잘못 분류된 학습패턴에 대한 적응도 계산시의 가중치를 높임으로서 입력공간의 넓은 영역에 분산되어 있는 패턴들에 대해서도 분류를 잘 하게 할 수 있다.

3.6. 서브패턴의 교체

서브패턴들의 적응도가 계산되고 나면 $|S|\times(1-|Xc|/|X|)$, 즉 서브패턴의 개수에 1-정확도를 곱한 값만큼의 하위 적응도의 서브패턴들을 새로 생성한 서브패턴들로 교체한다. 교체수가 1-정확도에 비례하도록 하는 이유는 현재의 정확도가 낮으면 앞으로 서브패턴 집합을 향상시켜야 할 소지가 많을 것이므로 많은 서브패턴들을 교체하도록 하고 정확도가 높으면 이미 충분히 좋은 서브패턴들을 가지고 있다고 가정할 수 있기 때문에 적은 수의 서브패턴만 교체하도록 하게 하기 위함이다.

또한 두 서브패턴이 동일하게 교체 기준점에 해당하는 적응도를 가지고 있고 둘 중 하나는 교체를 해야 하는 경우엔 길이가 더 긴 서브패턴을 교체하도록 한다. 오캄의 면도날(Occam's razor) 원리에 의하면 같은 성능이면 상대적으로 단순한 형태의 모델을 선호하는 것이 일반화 성능 차원에서 더 바람직하기 때문이다.

한편, 새로 생성되는 서브패턴의 길이 선택의 확률분포 D는 처음에는

균등분포였다가 반복문을 수행해 가면서 남아있는 서브패턴의 길이의 분포로 바뀌게 되는데 이를 통해 서브패턴의 길이가 적응도가 높은 길이로 선택 될 가능성을 높이게 된다. 주어진 학습패턴들에 따라 학습이 끝났을 시의 서브패턴들의 길이의 분포가 다르게 되는데 이는 각 학습패턴들의 분포의 차이를 반영하게 된다. 특정 클래스의 학습패턴들이 특정 공간에 밀집해 있는 경우에는 서브패턴의 길이가 길더라도 정확도가 높으면서 동시에 많은 학습패턴들을 만족시킬 수 있는 반면 학습패턴들이 입력 공간에서 넓게 분산되어 있는 경우엔 서브패턴의 길이가 짧아야지만 여러 학습패턴을 만족시킬 수 있다. 이 때 서브패턴의 길이가 짧으면 그만큼 넓은 공간을 커버하게 되고 다른 클래스의 패턴도 만족시킬 가능성이 커지므로 정확도는 떨어질 가능성이 있다. 즉, 각 서브패턴의 길이에 따라 학습모델의 일반성과 정확성이 달라지는 경향이 있음을 알 수 있다. 이 과정을 통해 현재 주어진 패턴을 분류하는데 적합한 길이의 서브패턴들이 생성될 수 있도록 할 수 있다.

또한 서브패턴의 결정 경계 선택의 확률분포 B도 처음에는 균등분포지만 반복문을 수행해 가면서 이전 단계에서 살아남은 결정 경계 형태의 비율에 해당하는 확률로 생성되는 서브패턴의 결정 경계가 정해진다. 이 과정을 통해 현재 주어진 패턴을 분류하는데 적절한 확률 분포로 서브패턴의 결정 경계가 정해지게 된다.

3.7. 수행 속도 및 소요 메모리

본 논문에서 제시하는 학습 알고리즘의 시간 복잡도는 각 서브패턴 별 적응도 계산 부분의 영향으로 $O(|X||S|)$, 즉 학습패턴의 수와 서브패턴의

수의 곱에 비례하게 된다. 그러므로 학습패턴의 수와 생성되는 서브패턴의 수가 증가함에 따라 수행 속도가 매우 느려질 수 있다. 속도 저하를 완화하기 위해 우선 서브패턴의 적응도 계산 시 어떤 학습패턴의 분류결과가 그 직전 반복문에서의 결과와 동일할 경우 적응도에 영향을 미치지 않으므로 해당 학습패턴은 무시하고 분류 결과가 다른 경우에만 처리하도록 할 수 있다. 알고리즘의 후반에 가면 학습패턴의 분류결과에 변화가 적게 발생하므로 적응도 계산에 필요한 시간을 줄일 수 있다.

또한 서브패턴의 유효성 여부는 한번만 계산되면 그 이후 해당 서브패턴이 살아 있는 한 동일한 결과를 가져오게 되므로 각 서브패턴마다 모든 학습패턴에 대한 유효성 여부를 비트 단위로 저장하게 되면 그 이후에는 유효 여부를 계산하지 않아도 되므로 수행 속도 향상을 볼 수 있다. 단, 이 경우 각 서브패턴 마다 학습패턴의 수만큼의 비트 공간을 추가로 가져야 하므로 공간 복잡도가 $O(|X||S|)$가 된다는 단점이 있다. 그러므로 이 방법은 메모리 공간이 충분하거나 학습패턴과 서브패턴의 수가 많지 않은 경우에만 유용하다.

위의 보완 알고리즘들 외에 각 서브 패턴의 적응도 계산이 다른 서브패턴에 영향을 받지 않고 비교적 구조가 간단하다는 점을 이용해서 전용 하드웨어를 만들어서 각 서브패턴 별 적응도 계산을 병렬처리로 한 번에 빠르게 하는 방법이 연구되고 있다 [22]. 하지만 병렬처리를 한다고 해도 서브패턴의 수가 매우 많은 경우 메모리 공간상의 문제로 인해 성능 향상에 제약이 있게 된다. 그러므로 서브패턴 전체를 임의로 생성 후 선택 및 교체 과정만 사용하기보다는 추가적으로 학습패턴의 특징에 따라 적절한 휴리스틱을 혼용하여 적은 수의 서브패턴만 가지고도 효율적으로 분류를 수행할 수 있도록 하는 것을 고려할 필요가 있다.

Ⅳ. 실험 결과

4.1. 실험 데이터

UCI Machine Learning Repository의 데이터 중 속성이 30개 이상, 레코드가 200개 이상인 데이터 4개를 골라서 CVLS의 분류 성능을 테스트해 보았다. 표 2는 선택된 데이터들에 대한 정보이다.

표 2. 실험 데이터

이름	속성 개수	레코드 개수
SONAR	60	208
SPECTF	44	학습: 80, 테스트: 187
Wisconsin Diagnostic Breast Cancer (WDBC)	31	569
IONOSPHERE	34	351

4.2. 실험 매개변수

주어진 패턴의 조건에 대한 서브패턴의 만족 여부를 판정할 때 적용되는 매개변수들인 유효 거리 r, 적응도 계산 가중치 α, 서브패턴의 개수를 다양한 값들로 설정하여 테스트 해보고 결과 차이를 비교해 본다. 특히 $r=2$, $\alpha=0.5$, 서브패턴의 개수는 학습패턴의 수의 50배로 설정한 것을 기준으로 알고리즘의 반복문이 수행됨에 따라 변화하는 테스트 정확도의 추이를 보고 일반적으로 분류 성능이 뛰어난 것으로 알려진 다른 분류기들과 비교해본다 [23]. 다른 분류기들의 Weka 3.5.7의 구현을 이용하여 실험하였다 [24].

지정한 매개변수들에 대해 총 10번의 실험을 수행하여 평균 정확도를 측정하였다. 이 때 SPECTF를 제외한 데이터에 대해서는 10 fold stratified cross validation을 적용하여 학습패턴, 테스트패턴을 구분하였다.

4.3. SONAR

그림 6은 SONAR에 대한 테스트 정확도의 변화를 나타낸다. Iteration은 수행된 반복문의 횟수, Accuracy는 분류 정확도를 나타낸다. 이를 통해 학습 알고리즘의 반복문이 수행될수록 분류 정확도가 향상되다가 어느 정도 이상 반복되면 수렴 되는 경향이 있음을 볼 수 있다.

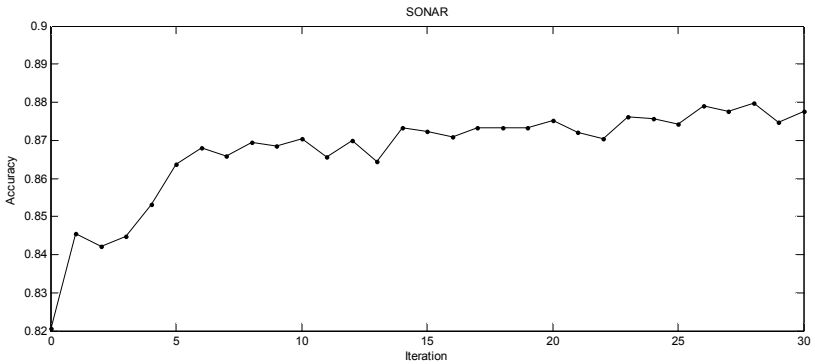

그림 6. SONAR 학습 중의 정확도 변화

표 3은 SONAR에 대한 테스트 정확도를 다른 분류기들과 비교한 결과이다. CVLS의 테스트 정확도가 다른 분류기들의 정확도 보다 뛰어남을 볼 수 있다.

표 3. SONAR에 대한 테스트 정확도 비교

분류기	평균 정확도 (%)	표준편차
CVLS	87.7404	1.798869
SVM (Pearson VII kernel [25])	86.29808	1.225735
*k*NN (*k*=1)	86.15385	0.871897
AdaBoost (반복횟수=50)	84.66347	1.330242
Random Forest (트리수=50)	84.18271	1.544634
Decision Tree (C4.5)	73.60576	2.618124
Naive Bayes	67.69233	1.082175

그림 7은 SONAR의 학습이 완료된 후의 서브패턴 길이의 빈도수를 나타낸 것이다. 서브패턴의 결정 경계는 Hypercube는 초입방체, Hypersphere는 초구체, Sum은 둘을 섞어 쓰는 경우이다. 결정 경계가 초구체인 서브패턴들이 초입방체인 서브패턴들 보다 많이 살아남았고 전체 서브패턴의 평균 길이는 17 정도이며 길이가 길수록 살아남는 수가 적음을 볼 수 있다. 이는 어떤 서브패턴의 모든 속성에 대해서 주어진 패턴의 모든 대응되는 속성과의 거리 조건을 만족시킨 경우에만 그 서브패턴이 유효한 것으로 판정되기 때문에 길이가 길수록 서브패턴이 유효하지 않게 될 가능성이 크게 되고 같은 적응도를 가진 두 서브패턴이 있으면 길이가 짧은 서브패턴을 선호하기 때문이다.

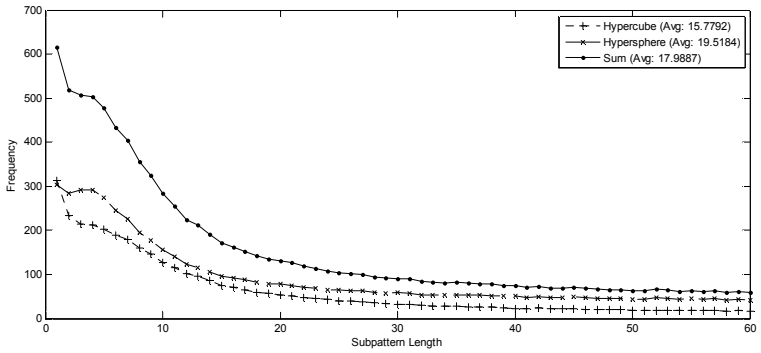

그림 7. SONAR 학습 완료 후 서브패턴 길이의 분포

그림 8은 SONAR에 대한 입력패턴이 서브패턴을 만족하는지 여부를 판별하는 (1)과 (2) 식의 기준 거리 r 과 분류 결정경계의 형태에 따른 정확도의 변화를 나타낸다. 결정 경계의 형태는 초입방체, 초구체, 같은 비율로 생성되는 초입방체와 초구체, 살아남은 서브패턴의 결정 경계 비율로 생성되는 초입방체와 초구체의 총 4가지로 정해진다. 이 때 r 이 너무 크면 지나치게 많은 서브패턴들이 유효하게 되어 분류에 관여할 수 있으므로 상관없는 서브패턴들까지 사용되게 될 가능성이 있고 너무 작으면 극소수의 서브패턴들만 분류에 관여할 수 있으므로 적절한 r 값을 지정하는 것이 중요하다. 그리고 결정 경계가 기존의 규칙기반 분류기들처럼 초입방체 형태로만 정해지거나 초구체 형태로만 정해지는 것 보다 서브패턴에 따라 두 가지 형태 중 한 가지를 사용하게 하고 또한 기존의 적응도에 따라 새로 생성되는 서브패턴 형태의 비율이 달라지게 하는 것이 실험적으로 가장 높은 수준의 정확도를 보임을 알 수 있다.

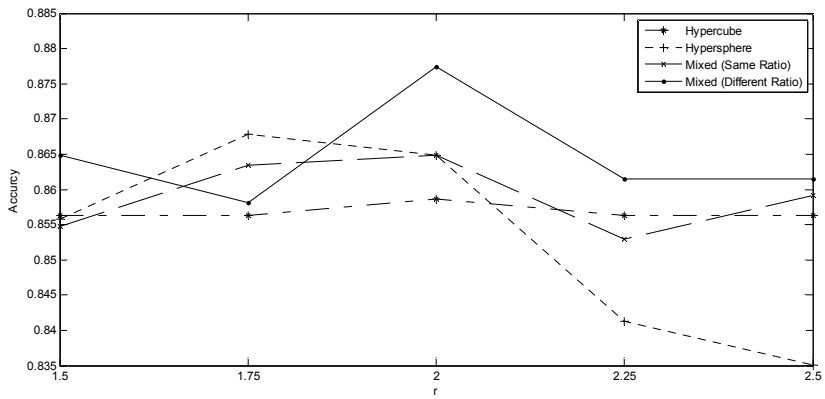

그림 8. SONAR 유효성 판별식의 r 과 결정경계 형태에 따른 정확도 차이

4.4. SPECTF

그림 9는 SPECTF에 대한 정확도의 변화를 나타낸다. SPECTF를 학습 시 초반에는 정확도가 매우 낮지만 몇 번의 반복만으로 상대적으로 높은 수준의 정확도로 수렴하는 것을 볼 수 있다.

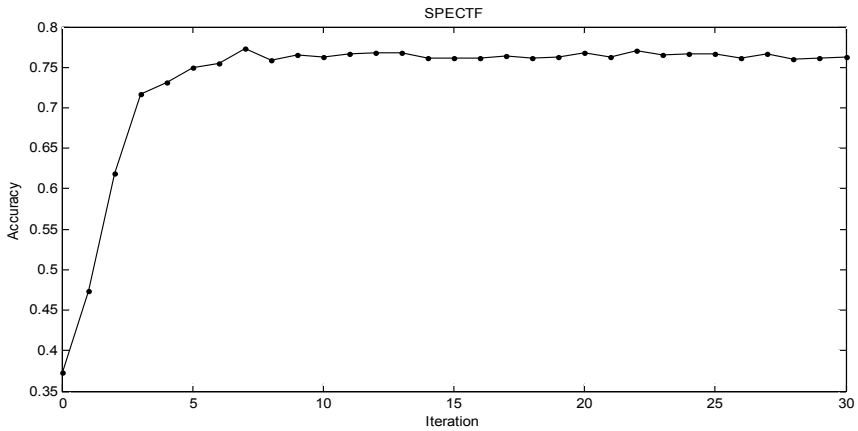

그림 9. SPECTF 학습 중의 정확도 변화

표 4는 SPECTF에 대한 정확도를 다른 분류기들과 비교한 결과이다. SPECTF에 대해 다른 분류기보다 높은 정확도로 테스트패턴을 분류할 수 있음을 볼 수 있다.

표 4. SPECTF에 대한 테스트 정확도 비교

분류기	평균 정확도 (%)
CVLS	**76.2567**
SVM (Pearson VII kernel)	74.3316
AdaBoost (반복횟수=50)	73.262
Random Forest (트리수=50)	71.6578
Naive Bayes	69.5187
Decision Tree (C4.5)	67.9144
*k*NN (*k*=1)	61.4973

그림 10은 SPECTF의 학습이 완료된 후의 서브패턴 길이의 분포를 나타낸 것으로 초입방체 형태 결정 경계를 갖는 서브패턴이 보다 많이 살아남았고 길이는 평균 5 초반 정도로 매우 짧은 경우가 대부분임을 볼 수 있다. 이는 SPECTF의 데이터들이 상대적으로 넓은 공간에 분산되어 있어서 서브패턴의 길이가 긴 경우에는 조건을 충족하여 적응도에 반영하게 되는 경우가 드물게 되기 때문에 발생하는 일로 추정된다.

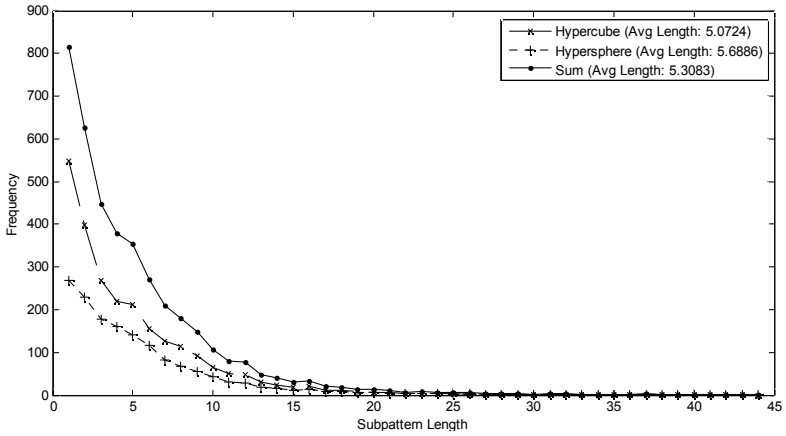

그림 10. SPECTF 학습 완료 후 서브패턴 길이의 분포

그림 11에서는 SPECTF에 대해 결정 경계가 초입방체 형태이거나 섞어서 쓰는 경우 정확도가 비슷하고 초구체 형태만 쓰는 경우에는 정확도가 떨어지게 됨을 볼 수 있다.

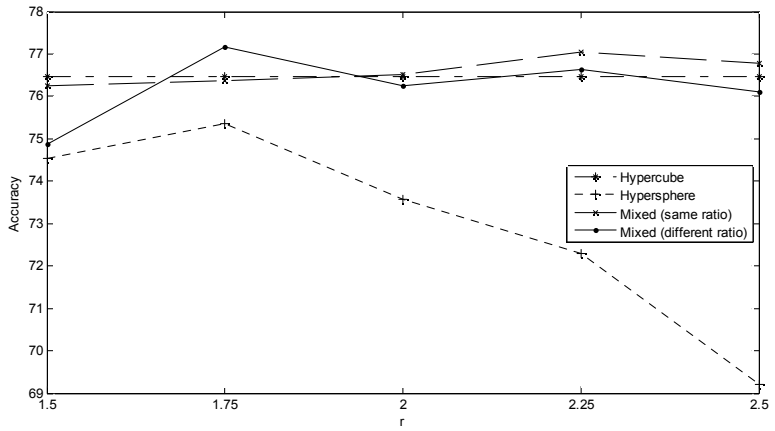

그림 11. SPECTF 유효성 판별식의 r 과 결정경계 형태에 따른 정확도 차이

4.5. IONOSPHERE

그림 12는 IONOSPHERE에 대한 정확도의 변화를 나타낸다. 5회 정도 반복이 수행될 시 93% 정도의 정확도로 수렴이 되는 것을 볼 수 있다.

표 5는 IONOPHERE 데이터에 대한 테스트 정확도를 다른 분류기들과

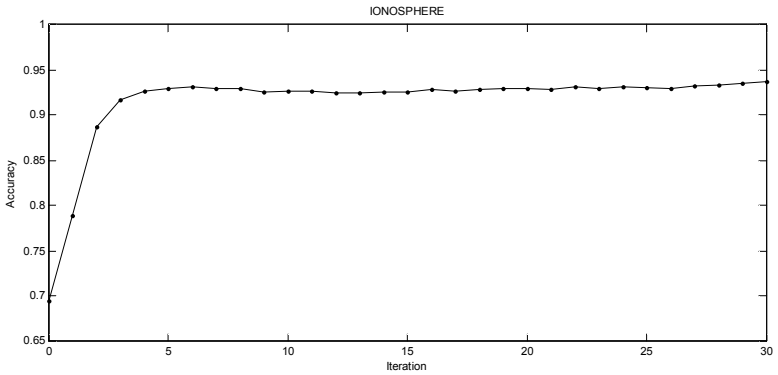

그림 12. IONOSPHERE 학습 중의 정확도 변화

비교한 결과이다. IONOSPHERE는 왠만한 분류기들의 정확도가 80%이상

나오는 것을 봐서 상대적으로 쉬운 데이터임을 알 수 있고 상대적으로 쉬운 데이터에 대해서는 CVLS의 장점이 크게 부각되지 않음을 볼 수 있다.

표 5. IONOSPHERE에 대한 테스트 정확도 비교

분류기	평균 정확도 (%)	표준편차
SVM (Pearson VII kernel)	94.47294	0.306258
AdaBoost (반복횟수=50)	93.84616	0.54056
CVLS	**93.6752**	**0.612517**
Random Forest (트리수=50)	93.56126	0.42893
Decision Tree (C4.5)	89.74360	1.433965
*k*NN (*k*=1)	87.09403	0.485167
Naive Bayes	82.16516	0.48791

그림 13은 IONOSPHERE의 학습이 완료된 후의 서브패턴 길이의 분포를 나타낸 것이다. IONOSPHERE의 경우에도 초입방체 결정 경계의 서브패턴들이 보다 많이 살아남음을 볼 수 있다.

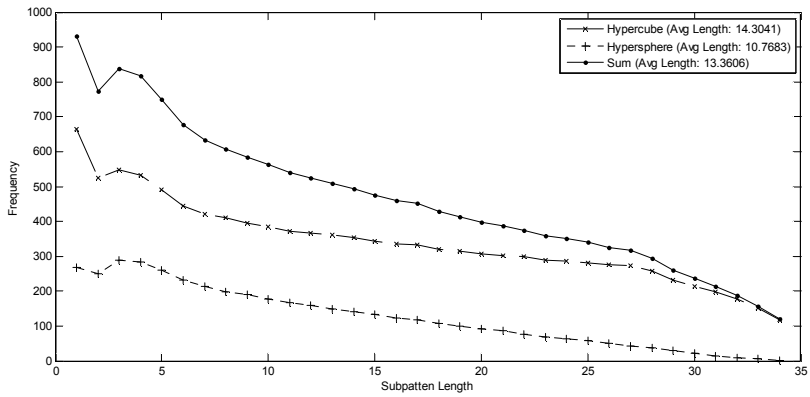

그림 13. IONOSPHERE 학습 완료 후 서브패턴 길이의 분포

그림 14에서는 IONOSPHERE의 경우 SPECTF처럼 서브패턴의 결정 경계가 초입방체로 이루어진 경우와 섞여서 쓰이는 경우가 비슷하고 초구체만 쓰는 경우에는 정확도가 뒤떨어지는 것을 볼 수 있다.

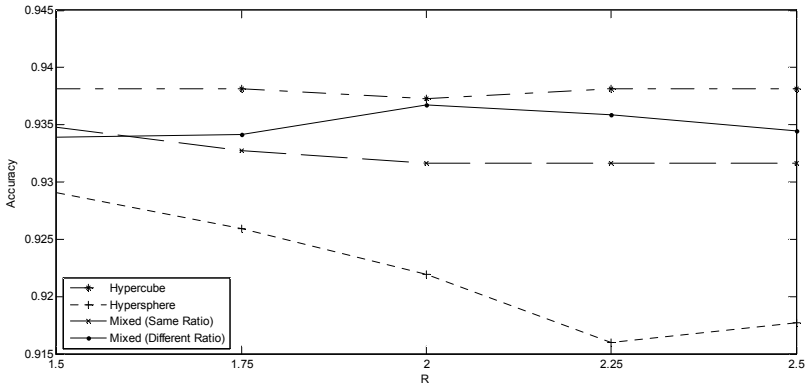

그림 14. IONOSPHERE 유효성 판별식의 r 과 결정경계 형태에 따른 정확도 차이

4.6. Wisconsin Diagnostic Breast Cancer (WDBC)

그림 15는 WDBC 데이터에 대한 테스트 정확도의 변화를 나타낸다. 5회 정도 반복이 된 후 대략 95% 후반 정도의 정확도로 수렴이 되는 것을 볼 수 있다.

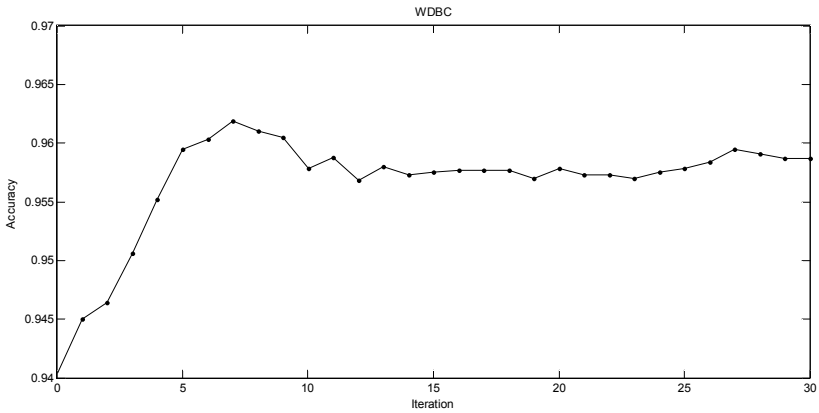

그림 15. WDBC 학습 중의 정확도 변화

표 6은 WDBC 데이터에 대한 테스트 정확도를 다른 분류기들과 비교한 결과이다. 이 데이터의 경우 모든 속성들이 서로 독립임을 가정하는 Naive Bayes만 사용해도 92%라는 높은 정확도가 나오는 상당히 쉬운 데이터임을 알 수 있다. IONOSPHERE와 마찬가지로 상대적으로 쉬운 데이터에 대해서는 CVLS의 장점이 부각되지 않으므로 정확도도 다른 분류기들과 비교 시 두드러지는 정확도를 보이지 않음을 볼 수 있다.

표 6. WDBC에 대한 테스트 정확도 비교

분류기	평균 정확도 (%)	표준편차
SVM (Pearson VII kernel)	97.4165	0.118609
AdaBoost (반복횟수=50)	96.8717	0.272256
Random Forest (트리수=50)	96.36204	0.36160
CVLS	**95.8699**	**0.372823**
*k*NN (*k*=1)	95.67662	0.30101
Decision Tree (C4.5)	93.00528	0.976077
Naive Bayes	92.7724	0.146670

그림 16은 WDBC의 학습이 완료된 후의 서브패턴 길이의 분포를 나타낸 것이다. 다른 데이터들의 경우 서브패턴의 길이가 길어질수록 학습 완료 후 살아남는 경우가 드물게 되었지만 WDBC의 경우에는 길이가 약 10이 될 때까지는 길이가 길수록 오히려 많은 서브패턴이 남고 20 이상이 되어야 그 비율이 줄어드는 것을 볼 수 있다. 이는 데이터들이 상대적으로 좁은 공간 안에 존재할 경우 서브패턴의 길이가 아주 짧으면 클래스가 다른 패턴들에 대해서도 유효한 것으로 간주되어 적응도가 낮아질 가능성이 높기 때문이다.

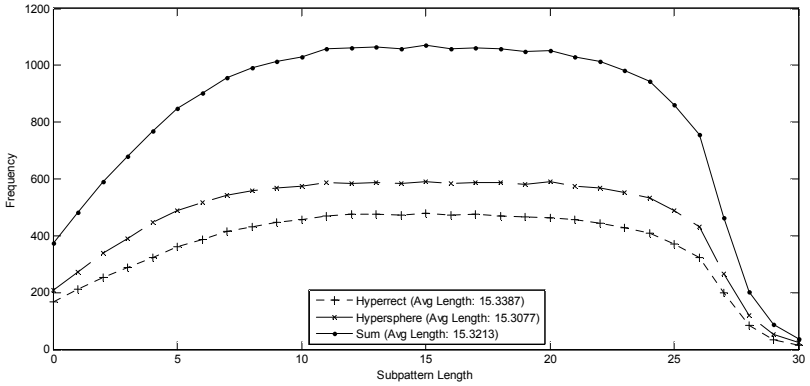

그림 16. WDBC 학습 완료 후 서브패턴 길이의 분포

그림 17을 통해 WDBC는 초구체 형태로 결정 경계가 구성될 때가 가장 정확도가 높고 초입방체로 구성이 될 경우 가장 정확도가 낮아짐을 볼 수 있다.

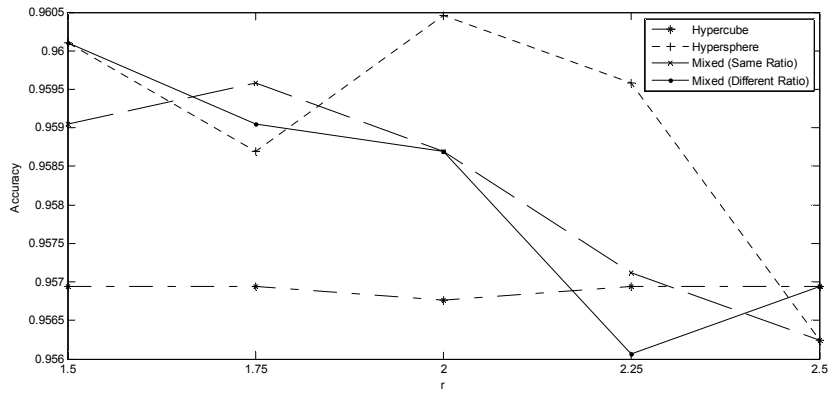

그림 17. WDBC 유효성 판별식의 r 과 결정경계 형태에 따른 정확도 차이

4.7. 실험 결과 분석

위의 4가지 데이터에 대한 실험을 통해 CVLS의 특징을 간접적으로 알 수 있다. IONOSPHERE나 WDBC는 평범한 분류기들을 사용했을 때도 정확도가 높게 나오는 상대적으로 쉬운 데이터인 것으로 보인다. 이러한 데이터의 경우 기존 분류기들을 적용해도 충분히 높은 정확도를 볼 수 있었고 CVLS의 경우 정확도가 상대적으로 평범하게 나타났다. 하지만 SONAR나 SPECTF 같은 데이터는 Naive Bayes로는 60%대 정도의 정확도밖에 볼 수 없는 상대적으로 어려운 데이터라고 할 수 있는데 이 데이터들의 경우에는 기존 분류기들로는 높은 정확도의 결과를 내는데 한계가 있는 반면 CVLS에서 서브패턴들로 다양한 형태의 결정 경계를 구성하는 것이 더 좋은 결과를 낼 수 있음을 볼 수 있다.

여기서 특히 Naive Bayes 분류기는 패턴 상의 모든 속성들이 서로 독립이라고 가정하고 분류를 수행하므로 Naive Bayes의 정확도가 낮다는 것은 패턴의 속성들 간의 상관관계가 높게 나타난다는 것을 의미한다고 할 수 있다. CVLS는 특성 상 입력 패턴이 주어져서 어떤 서브패턴이 조건을 만족 시키는지 여부를 따질 때 그 서브패턴 안에 있는 여러 속성들이 거리계산에 같이 사용될 수 있으므로 이들 속성간의 상관관계를 서브패턴으로 나타낼 수 있다는 장점이 있다. 일반적인 분류기들이 지역 최적화 알고리즘으로 지정되는 특정 속성들 간의 상관관계만 표현할 수 있는 반면 서브패턴에서는 속성들을 임의로 선택하게 되므로 보다 다양한 속성들 간의 상관관계를 나타낼 수 있다.

한편, 각 데이터에 따른 학습 완료 후 서브패턴 길이 분포의 추이를 볼 수 있는데 특정 길이의 서브패턴이 빈번하다면 서브패턴이 그 정도 수의 속성들을 가지고 있을 때 적응도가 높다는 것을 나타낸다. 일반적

으로 서브패턴의 길이에 따라 특정 영역에서의 정확도와 서브패턴이 커버할 수 있는 적용범위가 달라지는데 길이 분포를 통해 서브패턴 길이가 어느 정도가 되는 것이 이 데이터에 대해 분류를 수행 시 적절한지를 간접적으로 알 수 있다.

그림 18은 각 데이터 별 전체 서브패턴의 수에 따른 분류 정확도를 나타낸 것이다. 데이터에 따라 다르지만 전체 서브패턴의 수가 학습패턴 수의 5배에서 20배 정도가 되면, 즉 총 서브패턴 수로는 100대 중반에서 3000대 중반 사이 수 이상이 되면 더 이상 서브패턴이 늘어나도 정확도의 차이가 크지 않음을 볼 수 있다. 이를 통해 각 데이터별로 적절한 서브패턴의 수가 다르고 일정 수 이상의 서브패턴은 필요하지 않음을 알 수 있다. 일반적으로 속성의 개수가 많아서 문제 공간이 크거나 복잡한 결정 경계가 필요한 데이터는 보다 많은 서브패턴이 있어야 전체적인 결정 경계를 제대로 나타내고 일정 수준 이상의 분류 정확도를 보장할 수 있을 것이다.

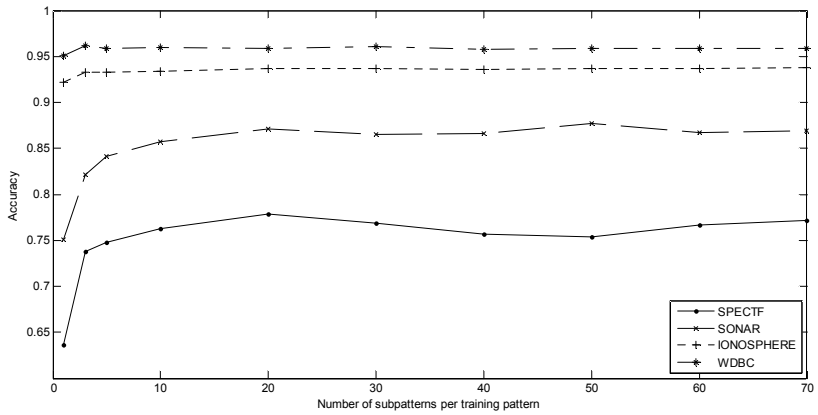

그림 18. 학습패턴 당 서브패턴의 수에 따른 테스트 정확도의 변화

Ⅴ. 결론 및 향후 과제

5.1. 결론

본 논문에서는 기존의 규칙 기반 분류기들의 단점인 지역 최적화만 가능한 규칙 생성 알고리즘, 결정 경계의 초입방체 형태 집합으로의 제약을 보완하는 서브패턴 기반의 학습 모델을 제시하였다. 이 모델은 학습 패턴의 부분집합인 서브패턴들을 임의로 생성하고 적응도 계산 후 교체 및 재생성 하는 단계를 거치기 때문에 다른 분류기들의 지역 최적화된 결과보다 높은 분류 정확도를 기대할 수 있고 적응도 계산 방법에 내재되어 있는 간접적인 부스팅 효과도 볼 수 있다. 그리고 서브패턴의 결정 경계 형태를 초입방체 뿐 아니라 입력패턴과의 유클리드 거리를 기준으로 하여 초구체로도 나타낼 수 있으므로 보다 다양한 결정 경계 형태가 가능하게 되어 분류 정확도를 향상시키는데 도움이 될 수 있다. 이외에도 서브패턴의 특성 때문에 학습패턴 중에 속성 값이 없는 패턴이 있는 경우에도 해당 속성에 영향 받지 않고 분류를 수행할 수 있고 다중 클래스 문제에도 동일하게 적용이 가능하다는 장점이 있다.

UCI Machine Learning Repository의 4가지 데이터에 대한 분류실험 결과 상대적으로 분류하기 쉬운 데이터에 대해서 다른 분류기들과 비교 시 상위권의 정확도를 보일 수 있고 특히 다른 분류기로는 높은 정확도를 보기 힘든 데이터에 대해 분류를 수행 시 경쟁력이 있는 모델임을 볼 수 있었다.

5.2. 향후 과제

본 논문에서 제시한 CVLS 모델은 임의로 생성된 많은 수의 서브패턴들을 사용할 경우 계산 비용의 증가로 인한 비효율성 문제가 생길 수 있다. 데이터에 따라 분류 정확도를 최대화하고자 할 때 필요한 서브패턴의 수가 다르므로 생성되는 서브패턴들의 수를 고정하는 것 보다 최적의 서브패턴 수를 찾아서 필요한 만큼의 서브패턴만 사용하도록 해야 효율적인 학습 및 분류를 수행할 수 있을 것이다. 특히 전역 최적해의 검색 가능성을 열어놓으면서 적절한 휴리스틱을 도입함으로서 적은 수의 서브패턴을 빠르게 찾는 방법을 향후 과제로 연구해야할 것이다. 이외에도 추가적인 결정 경계 형태의 도입이나 r, α 등 여러 가지 매개변수의 최적값을 자동으로 찾는 방법도 연구되어야 할 것이다.

참고문헌

[1] S. Kim, S.-J. Kim, and B.-T. Zhang, Evolving hypernetwork classifiers for microRNA expression profile analysis, *IEEE Congress on Evolutionary Computation (CEC 07)*, pp. 313-319, 2007

[2] J.-W. Ha, J.-H. Eom, S.-C. Kim, and B.-T. Zhang, Evolutionary hypernetwork models for aptamer-based cardiovascular disease diagnosis, *The Genetic and Evolutionary Computation Conference (GECCO 07)*, pp. 2709-2716, 2007.

[3] S. Kim, M.-O. Heo, and B.-T. Zhang, Text classifiers evolved on a simulated DNA computer, *IEEE Congress on Evolutionary Computation (CEC 06)*, pp. 9196-9202, 2006.

[4] J.-K. Kim and B.-T. Zhang, Evolving hypernetworks for pattern classification, *IEEE Congress on Evolutionary Computation (CEC 07)*, pp. 1856~1862, 2007.

[5] A. Wojna, Combination of Metric-Based and Rule-Based Classification, *Lecture Notes in Artificial Intelligence,* vol. 3641, pp. 501-511, 2005.

[6] A. Asuncion and D.J. Newman, UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science, 2007.

[7] J. Hong, I. Mozetic, and R. S. Michalski, AQ15: Incremental learning of attribute-based descriptions from examples, the method and user's guide, *In Report ISG 85-5, UIUCDCS-F-86-949,* Dept. Comp. Science, University of Illinois at Urbana-Champaign, 1986.

[8] P. Clark and T. Niblett, The CN2 induction algorithm, *Machine Learning*, 3:261-283, 1989.

[9] W. Cohen, Fast effective rule induction, *Proc. 1995 International Conference on Machine Learning (ICML 95)*, pp. 115-123, Tahoe City, CA, 1995.

[10] J. R. Quinlan and R. M. Cameron-Jones, FOIL: A midterm report, *Proc. 1993 European Conference on Machine Learning*, pp. 3-20, Vienna, Austria, 1993.

[11] B. Liu, W. Hsu, and Y. Ma, Integrating Classification and Association

Rule Mining, *Proc. 1998 International Conference on Knowledge Discovery and Data Mining (KDD 98)*, pp. 80-86, New York, NY, 1998.

[12] W. Li, J. Han, and J. Pei, CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules, *Proc. 2001 International Conference on Data Mining (ICDM 01)*, pp. 369-376, San Jose, CA, 2001.

[13] X. Yin and J. Han, CPAR: Classification based on Predictive Association Rules, *Proc. 2003 SIAM International Conference on Data Mining (SDM 03)*, pages 331-335, San Francisco, CA, 2003.

[14] F. Thabtah, P. Cowling, and Y. Peng, MCAR: Multi-class Classification based on Association Rule, *Proc. 3rd ACS/IEEE International Conference on Computer Systems and Applications*, Cairo, Egypt, pp. 33-39, 2005.

[15] R. Agrawal and R. Strikant, Fast algorithms for mining association rules in large databases, *Proc. 20th Intermatonal Conference on Very Large Databases*, pp. 487–499, Santiago, Chile, 1994.

[16] J. Han, J. Pei, and Y. Yin, Mining frequent patterns without candidate generation, *Proc. 2000 ACM-SIGMOD International Conference on Management of Data (SIGMOD 00)*, pages 1-12, Dallas, TX, 2000.

[17] J. R. Quinlan, Induction of decision trees, *Machine Learning*, 1:81-106, 1986.

[18] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993

[19] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Wadsworth International Group, 1984.

[20] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2006.

[21] Y. Freund and R. E. Schapire, A Short Introduction to Boosting, *Journal of Japanese Society for Artificial Intelligence,* 14(5):771-780, 1999.

[22] J.-K. Kim, B.S. Kim, O.H. Kwon, S.K. Hwang, J.-W. Ha, C.-H. Park, D.J. Chung, C.H. Lee, J. Park, and B.-T. Zhang, A DNA computing-inspired silicon chip for pattern recognition, *Preliminary Proceedings of the 13th International Meeting on DNA Computing (DNA13)*, pp. 373, 2007.

[23] R. Caruana and A. Niculescu-Mizil, An empirical comparision of supervised learning algorithms, *Proc. 23th International Conference on Machine Learning (ICML 06)*, pp. 161–168, 2006.

[24] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques, 2nd Edition*, Morgan Kaufmann, San Francisco, 2005.

[25] B. Uestuen, W.J. Melssen, and L.M.C. Buydens, Facilitating the application of Support Vector Regression by using a universal Pearson VII function based kernel, *Chemometrics and Intelligent Laboratory Systems,* vol. 81, pp. 29-40, 2006.

ABSTRACT

Joo-Kyung Kim
School of Computer Science and Engineering
The Graduate School
Seoul National University

Rules generated by the rule-based classifiers are easily interpreted and we can directly control those with lower relevancy to classification. However, most current rule-based classifiers can only show locally optimized solution and decision boundary is represented only as a set of hypercubes which are parallel to axes of input space.

In this paper, we introduce a subpattern-based classification model to overcome limits mentioned above. A subpattern is a subset of randomly selected attributes from each training pattern. After generated randomly, subpatterns will either remain or be substituted according to its fitness values. Because many combinations of subpatterns are tested in this process, the accuracy of the solution can be better than locally optimized solutions. Moreover, the proposed model allows both hypercubes and hyperspheres as the shapes of decision boundaries, and therefore it can represent various forms of decision boundaries. The proposed model demonstrated competitive accuracy with the UCI data, showing high accuracy with relatively high complexity issues especially.

Keywords : rule-based classification, decision boundary, subpattern, random sampling
Student Number : 2006-21175