프러스펙터의 분류 규칙 습득을 위한 유전자 알고리즘 기반 귀납적 학습 시스템

(A GA-based Inductive Learning System for Extracting the PROSPECTOR's Classification Rules)

김 영 준 †
(Yeong-Joon Kim)

요 약 주어진 사례의 집합으로부터 그 사례들을 분류할 수 있는 프러스펙터 규칙 유형의 분류 규칙들을 습득하는 학습 시스템을 유전자 알고리즘을 이용하여 구현하였다. 유전자 알고리즘을 이용한 학습 시스템의 구현에서 개체 집단은 규칙 집합으로 구성되고 규칙 집합은 교배, 돌연 변이, 역치 연산자 등의 유전 연산자를 이용하여 규칙 집합내의 규칙을 교환함으로써 새로운 지식을 생성한다. 본 논문에서는 구현된 학습 환경을 분류 규칙의 구문 형태와 의미, 개체 집단의 구조 및 유전 연산자의 구현 등을 중심으로 설명한다. 효율적인 돌연 변이 연산자의 구현을 위해 개발된 규칙 성능 평가 기법과 규칙 생성 기법은 소개하고 분류 성능을 향상시키기 위한 기법으로 다수의 규칙 집합을 이용하여 분류 시스템을 구축하기 위한 기법을 소개한다. 본 연구를 통해 구현된 학습 시스템의 성능을 다양한 사례 집합을 이용하여 평가하고 이를 신경망, 결정 트리 등과 비교하였다.

Abstract We have implemented an inductive learning system that learns PROSPECTOR-rule-style classification rules from sets of examples. In our approach, a genetic algorithm is used in which a population consists of rule-sets and rule-sets generate offspring through the exchange of rules relying on genetic operators such as crossover, mutation, and inversion operators. In this paper, we describe our learning environment centering on the syntactic structure and meaning of classification rules, the structure of a population, and the implementation of genetic operators. We also present a method to evaluate the performance of rules and a heuristic approach to generate rules, which are developed to implement mutation operators more efficiently. Moreover, a method to construct a classification system using multiple learned rule-sets to enhance the performance of a classification system is also explained. The performance of our learning system is compared with other learning algorithms, such as neural networks and decision tree algorithms, using various data sets.

1. 서 론

유전자 알고리즘은 주어진 문제에 대해 이진 문자를 이용하여 코딩 된 가능한 해들로 개체 집단을 생성한 후 개체 집단내의 구성원에 생물학적 진화 과정에서 볼 수 있는 유전 연산자들을 적용하여 새로운 개체 집단을 생성하는 과정을 반복하면서 주어진 문제의 최적 해를 찾는 탐색 알고리즘이다[1,2]. 유전자 알고리즘은 일반적인 탐색 문제 및 여러 최적화 문제 등의 해결에 널리 이용되고 있으며 기계학습 분야에서는 이를 이용한 학습 시스템의 구축에 관한 연구가 활발하게 진행되어 왔다.

본 연구의 목적은 유전자 알고리즘을 이용하여 주어진 사례의 집합으로부터 PROSPECTOR[3]에서 사용한 규칙의 형태를 취하는 분류 규칙들을 습득하는 학습 시스템을 구축하는 것이다. 유전자 알고리즘을 이용하여 생성 규칙들을 습득하기 위한 학습 시스템의 개발에 관한 연구는 이미 오래 전부터 진행되어 왔으나 대부분의 연구는 참과 거짓의 두 진리 값을 갖는 생성 규칙들을 습득하는 학습 시스템의 연구에 관한 것이었으며[4, 5, 6, 7], 최근에는 퍼지 컨트롤러의 구현을 위한 퍼지 규칙의 습득 및 퍼지 규칙에 기반을 둔 분류 시스템의 구

· 본 연구는 2000학년도 상명대학교 자연과학연구소 연구비지원에 의해 이루어졌음.
† 비 회 원 : 상명대학교 소프트웨어학부 교수
yjkim@sangmyung.ac.kr
논문접수 · 2000년 9월 1일
심사완료 · 2001년 8월 11일

축 등에 유전자 알고리즘을 이용하기 위한 연구가 활발히 진행되고 있다. Cooper와 Vidal[8], Homaifar와 McCormick[9], Chiang et al.[10] 등은 퍼지 컨트롤러의 설계를 위한 퍼지 소속함수와 퍼지 규칙의 습득에 유전자 알고리즘을 이용하였고 Yuan과 Zhaung[11]은 이산적인 속성 값을 갖는 사례들을 분류하기 위한 퍼지 규칙의 습득에 유전자 알고리즘을 이용하였다. Ishibuchi et al.[12, 13]은 연속적인 속성값을 갖는 사례들을 분류하기 위한 퍼지 규칙을 습득하는 시스템을 유전자 알고리즘을 이용하여 구현하였다.

비록 퍼지 컨트롤러의 구축에 유전자 알고리즘을 이용하기 위한 연구는 기존의 두 진리 값을 갖는 생성규칙에서 벗어난 다치 값에 기반을 둔 퍼지 규칙의 습득에 유전자 알고리즘을 이용하였다는 점에서 본 연구와 유사하나 그들이 습득하고자 하는 퍼지 규칙은 움직이는 물체의 균형을 유지하기 위한 기계적인 작동이나 수위를 조절하기 위한 밸브의 조작 등 동적인 환경하에서 시스템을 통제하기 위한 규칙들을 습득하기 위한 것으로 본 연구를 통해 습득하고자 하는 규칙과 같이 주어진 사례의 속성 값에 따라 그 사례가 특정 클래스에 속할 가능성을 제공하는 분석적이며 통계적, 확률적 정보를 다루는 규칙의 습득과는 다소 차이가 있다. 퍼지 규칙을 이용하여 결정/진단 시스템을 구현할 수 있으나 이러한 시스템의 구축에는 통계적, 확률적 정보를 다루는데 적합한 규칙을 이용하는 것이 더 타당하리라 사려되며 이러한 관점에서 볼 때 본 연구에서 구축하고자 하는 학습 시스템은 기존의 퍼지 규칙의 습득을 위한 학습 시스템의 구축과는 다소 차이가 있다.

본 연구를 통해 구현된 유전자 알고리즘 기반 학습 환경에서 개체 집단은 일정 수의 규칙 집합으로 구성되고 각각의 규칙 집합은 임의의 수의 분류 규칙으로 구성된다. 학습 과정에서는 현재의 개체 집단에 유전 연산자를 적용하여 새로운 개체 집단을 생성하는 과정을 만족할 만한 규칙 집합을 습득할 때까지 반복한다. 개체 집단 구성원인 규칙 집합의 적합도는 주어진 사례 집합에 대한 분류의 정확도로 평가한다. 본 논문에서는 구현된 학습 환경을 규칙의 구문 형태와 의미, 개체 집단 및 유전 연산자의 구현 등을 중심으로 설명한다. 유전자 알고리즘으로 하여금 탐색을 좀 더 효율적으로 수행할 수 있도록 하기 위해 연구한 규칙의 성능 평가 기법과 분류 작업에 적합한 규칙을 생성하기 위한 기법을 소개하고 분류 시스템의 성능을 향상시키기 위해 다수의 규칙 집합을 이용하여 분류 시스템을 구축하기 위한 기법을 소개한다. 또한 본 연구를 통해 구현된 학습 시스템의 성능을 다양한 사례 집합을 이용하여 평가하고 이를 신경망, 결정 트리 등과 비교하였다.

본 논문의 구성은 다음과 같다. 2장에서는 유전자 알고리즘을 이용하여 습득하고자 하는 규칙의 구문 형태 및 의미를 간략히 설명하고, 3장에서는 유전자 알고리즘을 이용한 학습 시스템의 구현을 자세히 설명한다. 4장에서는 규칙의 성능을 평가하기 위한 기법과 적절한 규칙을 생성하기 위한 기법을 설명하고 이를 이용한 돌연변이 연산자의 구현을 설명한다. 5장에서는 학습 시스템의 성능을 다양한 사례의 집합을 이용하여 평가하였다. 6장에서는 다수의 규칙 집합을 이용하여 분류 시스템을 구축하기 위한 기법을 소개하고 본 연구를 통해 구축된 학습 시스템의 성능을 신경망, 결정 트리 등을 이용하여 구축된 분류 시스템의 성능과 비교하였다. 7장에서 결론을 맺는다.

2. 규칙의 구문 형태 및 의미

클래스 C_1, C_2, ..., C_m으로부터 습득된 사례의 집합에서 각각의 사례는 속성 A_1, A_2, ..., A_n에 대한 값 a_1, a_2, ..., a_n과 그 사례가 속한 클래스 C_k로 구성된 리스트, (a_1, a_2, ..., a_n, C_k)의 형태로 표현된다 ($1 \le k \le m$). 속성 A_i에 대한 값 a_i는 사례가 A_i에 대해 실제 값 a를 가질 때 이를 사례 집합내의 다른 사례들의 A_i에 대한 값들과의 상대적 크기에 따라 식 (1)을 이용하여 0과 1 사이의 값으로 정규화 한 것이다.

$$a_i = \frac{A_i\text{에 대해 a보다 작은 값을 갖는 사례의 수}}{A_i\text{에 대해 값 a와 다른 값을 갖는 사례의 수}} \quad (1)$$

식 (1)에서 서로 다른 속성 값의 개수가 아닌 사례의 수로 정규화를 한 이유는 정규화 한 결과에 같은 값을 갖는 사례들의 수를 반영하기 위한 것이고 전체 사례수로 분모를 취하지 않은 이유는 속성 A_i에 대해 가장 큰 값을 갖는 사례들이 정규화 된 값으로 1을 갖도록 하기 위한 것이다.

학습 시스템은 주어진 사례의 집합으로부터 PROSPECTOR[3]에서 사용한 "If E then C with S=s, N=n"형태의 규칙들을 습득한다. 이들 습득된 규칙들을 이용하여 구축된 분류 시스템에서 각각의 규칙은 조건절(즉, E)에서 고려하는 속성에 대해 사례가 갖고 있는 값에 따라 그 사례가 클래스 C에 속할 가능성에 대한 승수를 S와 N 사이의 값으로 제공한다. 규칙은 사례가 규칙의 조건절을 완전하게 만족하면(즉, P(E′)=1) S의 값을, 만족 시키지 않을 때에는(즉, P(E′)=0) N의 값을, $0 < P(E') < 1$인 경우에는 P(E′)의 값에 비례하여 S와 N 사이의 값을 제공한다. 분류 시스템은 사례가 각

각의 클래스 C_k에 속할 사전 가능성 $O(C_k)$에 C_k를 결론절에서 참조하는 규칙들이 제공하는 승수를 곱하여 사후 가능성 $O(C_k')$를 구한 후 사후 가능성이 가장 높은 클래스를 주어진 사례가 속한 클래스로 선택한다. 사전 가능성 $O(C_k)$는 사례가 C_k에 속할 확률 $P(C_k)$로부터 PROSPECTOR에서 쓰인 방법에 따라 식 $O(C_k) = P(C_k)/P(\neg C_k)=P(C_k)/(1-P(C_k))$을 이용하여 구하고 $P(C_k)$는 주어진 사례 집합 내에서 C_k에 속하는 사례의 비율로부터 구한다.

분류 시스템이 사례를 분류하는 과정은 다음과 같다:

1. 각각의 클래스 C_k에 대해 사례 집합에서 C_k에 속한 사례의 비율에 따라 $P(C_k)$를 구한 후 사전 가능성 $O(C_k)=P(C_k)/(1 - P(C_k))$를 구한다.
2. C_k를 결론절에서 참조하는 규칙

   (r_1) If E_1 then C_k with $S=s_1$, $N=n_1$

   ...

   (r_p) If E_p then C_k with $S=s_p$, $N=n_p$

들이 제공하는 C_k에 대한 승수

$$\lambda_i=\frac{O(C_k \mid E_i')}{O(C_k)} \text{ for } i=1,...p$$

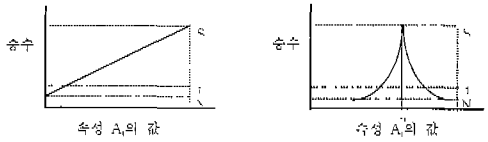
를 이용하여 사후 가능성

$$O(C_k')=O(C_k \mid E_1' \wedge ... \wedge E_p')=O(C_k)\times \prod_{i=1}^{p}\lambda_i$$

을 구한다.

3. 사후 가능성이 가장 큰 클래스를 주어진 사례가 속한 클래스로 선택한다.

학습 시스템은 사례의 집합으로부터 두 가지 타입의 규칙들로 이루어진 규칙 집합을 습득한다. 규칙의 타입 중 하나는 "If is-high(A_i) then C_k with S=s, N=n"로 이 타입의 규칙은 사례가 A_i에 대해 갖고 있는 값의 상대적인 크기에 따라 C_k에 속할 가능성에 대한 승수를 S와 N 사이의 값으로 제공한다. 즉, 이 형태의 규칙은 사례가 사례 집합 내에서 A_i에 대해 가장 큰 값을 가지면 ($P(E')=P(\text{is-high}(A_i)')=1$) S의 값을, 가장 작은 값을 가지면 ($P(E')=0$) N을, 그 외의 경우에는(즉, $0 < P(E') < 1$ 인 경우) $\lambda = S* P(E') + N*(1 - P(E'))$의 승수를 제공한다 (그림 1 (a) 참조). 다른 하나는 속성 A_i의 값이 어떤 특정 값 a에 근사한 정도에 따라 클래스 C_k에 대해 S와 N사이의 값을 제공하는 "If is-close(A_i, a) then C_k with S=s, N=n"의 형태의 규칙으로 이 형태의 규칙은 조건절에서 참조하는 속성에 대해 사례가 갖고 있는 속성 값 a'로부터 $P(E')=P(\text{is-close}(A_i, a)') = \max(0, 1 - 2 * \sqrt{|a'-a|})$를 구한 후 $\lambda = S*P(E')+ N*(1-P(E'))$의 승수를 제공한다 (그림 1 (b) 참조). 학습 시스템은 주어진 사례의 집합에 대해 그 사례들을 분류하기 위한 규칙 집합(즉 주어진 사례들을 분류하기 위해 필요한 속성들을 적절히 고려한 is-high 규칙과 is-close 규칙들)을 각각의 규칙에 필요한 s, n, 상수 a의 값(is-close 규칙에 한 함)과 함께 유전자 알고리즘을 이용하여 습득하는 것이다.



(a) is-high 규칙 (b) is-close 규칙

그림 1 승수의 계산

그림 2는 학습 시스템이 세가지 종류 C_1, C_2, C_3의 사례들로 구성된 사례 집합으로부터 습득한 규칙 집합의 예를 보인 것이다. 규칙 집합은 주어진 사례가 속성 A_1, A_2, A_3 및 A_4에 대해 갖고 있는 값에 따라 그 사례가 어느 클래스에 속하는지를 분류해 낼 수 있는 규칙들로 구성되어 있다.

```
If is-high(A4) then C1 with S=1944, N=0.016
If is-high(A2) then C2 with S=0.002, N=3139980
If is-close-to(A1,0.0198) then C2 with S=0.8, N=534
If is-high(A3) then C1 with S=0.42, N=4.9e+12
If is close-to(A1,0.774) then C2 with S=3.2e+6, N=0.003
If is-high(A1) then C1 with S=9.9e+18, N=0.21
```

그림 2 습득된 규칙 집합의 예

위의 규칙 집합에서 첫 번째 규칙은 A_4의 값이 클수록 주어진 사례가 C_1에 속할 가능성이 높은 것으로 간주하여 C_1의 가능성을 향상 시킬 수 있도록 S에 가까운 승수를 제공하고 A_4의 값이 작을 수록 C_1일 가능성을 감소시키도록 N에 가까운 승수를 제공한다. 세 번째 규칙은 A_1의 값이 0.0198에 가까울수록 주어진 사례가 C_2일 가능성에 대해 S에 가까운 승수를 제공하여 그 가능성을 감소시키며 0.0198에 멀어질수록 N에 가까운 승수를 제공하여 C_2의 가능성을 증가 시켜준다. 주어진 사례에 대해 각각의 규칙들은 그 규칙이 참조하는 클래스에 대해 S와 N 사이의 승수를 제공하고, 분류 시스템은 이들 규칙들이 제공한 승수를 사전 가능성에 곱하여 사후 가능성을 구한 후 사후 가능성이 가장 큰 클래스를 선택하는 것이다.

3. 유전자 알고리즘을 이용한 학습 시스템의 구현

유전자 알고리즘을 이용한 학습 시스템의 구현에서 개체 집단은 일정 수의 규칙 집합으로 구성되며, 규칙 집합은 임의의 수의 is-high규칙과 is-close규칙들로 구성된다. 즉, 개체 집단의 구성원은 임의의 수의 is-high 규칙과 is-close규칙들로 이루어진 규칙의 집합이다. 규칙은 규칙의 타입, 기준치 a의 값(is-close 규칙인 경우), 조건절에서 참조한 속성, 결론절에서 참조할 클래스, S 와 N의 값을 저장하기 위한 6개의 필드로 구성되며 is-high규칙은 0, is-close 규칙은 1로 표현되어 규칙의 타입 필드에 저장되고, 사례 집합내의 속성의 수를 n, 클래스의 수를 m이라 하면, 규칙이 참조할 속성과 클래스는 각각 1 .. n , 1 .. m의 정수 중 하나로 표현되어 속성 및 클래스 필드에 저장된다. S와 N 그리고 기준치 a는 실수 값을 갖는다. 그림 3은 개체 집단의 구성원인 규칙 집합과 규칙의 구조를 보인 것이다.

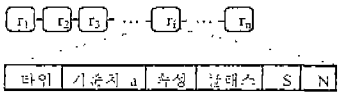
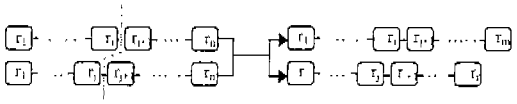
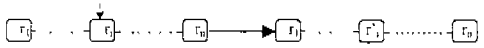


그림 3 규칙 집합과 규칙의 구조

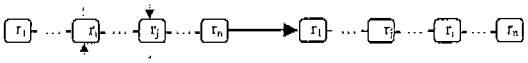
초기의 개체 집단은 난수 발생기를 이용하여 일정 수의 규칙 집합을 생성함으로써 얻어진다. 각각의 규칙 집합은 우선 난수 발생기를 이용하여 규칙의 수를 정한 후 해당되는 수 만큼의 규칙을 난수 발생기를 이용하여 규칙의 타입, 참조한 속성, 클래스, S, N 및 a의 값을 정하는 과정을 통해 생성된다. 계속되는 진화 과정에서는 주어진 개체 집단에서 새로운 개체 집단을 생성하는 과정을 만족할 만한 규칙 집합이 얻어질 때 까지 반복하게 되는데 이 과정에서 교배 연산자, 돌연변이 연산자, 역치 연산자 등의 유전 연산자가 이용되었다. 교배 연산자는 규칙 집합의 적합도에 비례하여 선택된 두 개의 규칙 집합내의 규칙들의 일부를 교환하여 새로운 규칙 집합을 생성한다 (그림 4 (a)). 돌연 변이 연산자는 규칙 집합내의 하나의 규칙을 새로 생성된 임의의 규칙으로 대체 함으로써 새로운 규칙 집합을 생성한다 (그림 4 (b)). 역치 연산자는 규칙 집합내의 두 개의 규칙을 선택하여 이의 위치를 교환해 줌으로써 새로운 규칙 집합을 생성한다 (그림 4 (c)). 역치 연산자의 역할은 서로 연관성이 있는 규칙들을 인접한 위치에 놓이게 함으로써 교배 연산자를 이용하여 새로운 규칙 집합을 생성할 때 이들 연관된 규칙들이 분리 될 가능성을 감소시켜 가능한 한 같은 규칙 집합 내에 있도록 하기 위해 사용되었다.



(a) 교배 연산자



(b) 돌연 변이 연산자



(c) 역치 연산자

그림 4 유전 연산자

규칙 집합의 적합도는 규칙 집합내의 규칙들로 주어진 사례들을 어느 정도 정확하게 분류 할 수 있는가 하는 분류의 정확도를 이용하여 평가한다. 즉 주어진 사례의 집합에 100개의 사례가 있을 때 이들 사례를 규칙 집합내의 규칙들을 이용하여 분류를 했을 때 80개를 정확하게 분류 하였다면 그 규칙 집합의 해로써의 적합도는 80/100=0.8이 되는 것이다.[1] 그림 5는 주어진 개체 집단 G(t)에서 새로운 개체 집단 G(t+1)을 생성하기 위한 알고리즘을 보인 것이다.

```
next-generation(t):=
INSERT best-member-of(G(t)) into G(t+1);
  DO { Select two members RS1 and RS2(≠RS1) using
roulette-wheel;
  If should-make-crossover( )
      then crossover(RS1,RS2,RSnew1, RSnew2)
      else {RSnew1 := RS1; RSnew2 := RS2}
  If should-have-mutation( ) then RSnew1:= mutate(RSnew1);
  If should-have-mutation( ) then RSnew2:= mutate(RSnew2);
  If should-have-inversion( ) then RSnew1:= invert(RSnew1);
  If should-have-inversion( ) then RSnew2:= invert(RSnew2);
  Insert RSnew1, RSnew2 into G(t+1); }
  UNTIL population of G(t+1) is complete;
}}}}
```

그림 5 새로운 개체 집단의 생성

그림 5에서 함수 best-...는 현재의 개체 집단에서 최적의 규칙 집합을 다음 개체 집단에 포함 시키기 위한

1) 실제 학습 시스템의 구현에서는 분류의 정확도를 규칙 집합내의 규칙의 수에 비례하여 감소시키도록 변형된 형태의 적합도 함수를 이용하였으며 이에 대한 보다 자세한 설명을 5장에서 하였음.

함수이다. *should-...* 함수는 참과 거짓을 반환하는 함수로 선택된 규칙 집합에 돌연 변이나 교배 연산자, 역치 연산자를 주어진 확률에 따라 적용하기 위해 사용한다. *crossover*함수는 두 개의 규칙 집합에 교배 연산자를 적용하여 새로운 규칙 집합을 생성한 후 이를 반환하는 함수이다. *mutate*와 *invert*는 규칙 집합에 돌연변이 연산자, 역치 연산자를 적용하여 그 결과를 반환한다.

4. 돌연 변이 연산자의 구현

3장에서 기술한 학습 시스템의 구현에서 돌연 변이 연산자의 역할은 규칙 집합에서 하나의 규칙을 선택하여 이를 새로운 규칙으로 교체하는 것이다. 이러한 역할을 담당할 돌연 변이 연산자를 구현하는 가장 간단한 방법은 난수 발생기를 이용하여 규칙 집합내의 모든 규칙에 대해 동일한 확률로 대체될 규칙을 선택하고 이를 난수 발생기를 이용하여 생성된 임의의 새로운 규칙으로 교체하도록 하는 것이다. 그러나 이러한 방법에 따라 구현된 돌연 변이 연산자는 주어진 사례들의 분류에 적합한 규칙을 적합하지 않은 규칙과 동일한 확률로 선택하여 이를 규칙 집합에서 필요로 하는 규칙이 아닌 임의의 규칙으로 대체함으로써 학습이 진행될수록 탐색의 효율성을 저하시킬 가능성이 있다. 그러므로 유전자 알고리즘으로 하여금 탐색 공간을 효율적으로 탐색하게 하기 위해서는 규칙 집합의 분류 성능을 저하시키는 규칙을 선택하여 이를 규칙 집합에 필요한 새로운 규칙으로 대체하는 효율적인 돌연변이 연산자가 필요하다. 이를 위해 규칙이 분류 작업에 기여하는 정도를 평가한 후 분류 작업에 기여하지 못하는 규칙을 좀 더 높은 확률로 선택하여 제거하고 규칙 집합에서 필요로 하는 규칙을 생성하여 삽입하도록 돌연 변이 연산자를 구현하였다.

4.1 규칙의 성능 평가

돌연 변이 연산자로 하여금 분류 작업을 수행하는데 부적합한 규칙을 그 부적합한 정도에 비례하여 선택하여 이를 새로운 규칙으로 대체하도록 하려면 규칙이 주어진 분류 작업을 수행하는데 어느 정도 적합한가를 수치적으로 평가할 수 있는 기법이 필요하다. 우리는 직관적으로 주어진 사례에 대한 분류 작업을 수행하기에 적합한 규칙과 부적합한 규칙을 구별할 수 있다. 만약 어떤 규칙이 사례가 속한 클래스에 대해 1 보다 큰 승수를 제공하여 사후 가능성을 증가 시키거나 사례가 속하지 않은 클래스에 대해 1보다 작은 승수를 제공하여 사후 가능성을 감소시킨다면 그 규칙은 그 사례를 분류하는데 적합한 규칙이라 할 수 있다. 이와 반대로 만일 규칙이 사례가 속한 클래스에 대해 사후 가능성을 감소시키거나 혹은 사례가 속하지 않은 클래스에 대해 사후 가능성을 증가 시킨다면 그 규칙은 주어진 사례의 분류 작업을 수행하는데 적합한 규칙이라 할 수 없다.

이러한 직관적인 고찰에 따라 사례에 대한 분류 결과에 기여한 정도에 비례하여 보상 또는 벌점을 부과하는 규칙 평가 함수를 개발하였다. 규칙 평가 함수는 만약 규칙 집합이 주어진 사례를 올바르게 분류하면 분류의 결과에 기여한 모든 규칙에 대해 보상을 한다. 좀 더 명확하게 설명하면 만약 규칙 집합이 주어진 사례를 올바르게 분류하면 선택된 클래스에 대해 1보다 큰 승수를 제공한 규칙과 다른 클래스에 대해 1보다 작은 승수를 제공한 규칙에 대해 그들 규칙이 제공한 승수에 비례하여 보상을 한다. 반면에 만약 규칙 집합이 주어진 사례를 잘 못 분류한 경우에는 그 잘못된 결정에 기여한 모든 규칙 즉 잘못된 결정에 대해 1보다 큰 승수를 제시한 규칙과 사례가 속한 클래스에 대해 1보다 작은 승수를 제시한 규칙에 대해 벌점을 부과한다. 이 보상과 벌점에 대한 결과를 사례 집합내의 모든 사례에 대해 취합하면 그 결과는 주어진 분류 작업에 대한 규칙의 적합도를 나타낸다고 볼 수 있다. 취합한 결과가 클 수록 좀 더 적합한 규칙인 반면에 그 결과가 작을수록 부적합한 규칙이라 할 수 있다.

주어진 사례에 대한 분류 결과에 따라 규칙 평가 함수 f는 보상 또는 벌점을 다음과 같이 분배 한다:

λ_r을 규칙 r이 주어진 사례에 대해 제공하는 승수, G_k를 클래스 C_k에 대해 1보다 큰 승수를 제공하는 규칙의 집합, L_k를 1보다 작은 승수를 제공하는 규칙의 집합, O_k를 C_k가 아닌 다른 클래스에 대해 1보다 작은 승수를 제공하는 규칙의 집합으로 정의하면

Case 1: 분류 시스템이 클래스 C_k에 속하는 사례에 대해 클래스 C_k를 선택한 경우

$$f(r) = \begin{cases} \{|G_k|/(|G_k| + |O_k|)\} * (\lambda_r / \Sigma_{r' \in G_k} \lambda_{r'}) & \text{for } r \in G_k \\ \{|O_k|/(|G_k| + |O_k|)\} * \{(1/\lambda_r) / \Sigma_{r' \in O_k} (1/\lambda_{r'})\} & \text{for } r \in O_k \\ 0 & \text{others} \end{cases}$$

Case 2: 분류 시스템이 클래스 C_k에 속하는 사례에 대해 클래스 C_j를 선택한 경우

$$f(r) = \begin{cases} -\{|G_j|/(|G_j| + |L_k|)\} * (\lambda_r / \Sigma_{r \in G_j} \lambda_{r'}) & \text{for } r \in G_j \\ -\{|L_k|/(|G_k| + |L_k|)\} * \{(1/\lambda_r) / \Sigma_{r \in L_k} (1/\lambda_r)\} & \text{for } r \in L_k \\ 0 & \text{others} \end{cases}$$

분류 결과에 대한 전체 보상액 또는 벌점은 각각 1, -1이다. 함수 f는 사례를 올바르게 분류한 경우 우선 사

례가 속한 클래스에 대해 1보다 큰 승수를 제시한 규칙들의 집합과 다른 클래스에 대해 1보다 작은 승수를 제시한 규칙들의 집합에 대해 집합내의 규칙의 수에 비례하여 주어진 단위 보상액 1을 분할한 후 이를 다시 각각의 집합내의 규칙에 그 기여도에 따라 분배한다. 사례를 잘 못 분류한 경우에는 전체 벌점 -1을 잘 못 선택한 클래스에 대해 1보다 큰 승수를 제시한 규칙의 집합과 사례가 속한 클래스에 대해 1보다 작은 승수를 제공한 규칙의 집합에 그 집합의 크기에 따라 분할한 후 이를 집합내의 규칙에 기여도에 따라 적절히 분배한다.

함수 f가 보상 또는 벌점을 분배하는 과정을 예를 들어 설명하면 다음과 같다. 표 1은 세 개의 규칙 r_1, r_2, r_3로 구성된 분류 시스템을 이용하여 사례 t_1, t_2, t_3를 분류한 결과를 보인 것이라 가정하자. 표 1에서 '클래스'는 사례가 속한 클래스, '결과'는 사례에 대한 분류 결과, λ_r은 주어진 사례에 대해 규칙 r이 제공한 승수를 보인 것이다. 규칙 r_1, r_2, r_3가 각각 클래스 C_1, C_2, C_3를 참조한다는 가정하에 규칙 평가 함수를 이용하여 주어진 사례 t_1, t_2, t_3에 대한 분류 성능을 평가하면 다음과 같다.

표 1 사례에 대한 분류 결과

	t_1	t_2	t_3
클래스	C_1	C_2	C_3
λ_{r1}	1.80	2.68	2.68
λ_{r2}	0.80	37.14	0.80
λ_{r3}	0.69	67.88	25.89
결과	C_1	C_2	C_3

첫 번째 사례 t_1에 대해 분류 시스템은 t_1이 속한 클래스로 C_1을 선택하였으므로 분류 시스템은 주어진 사례를 올바르게 분류 하였다. 따라서 집합 G_1에 속하는 규칙, 즉 사례 t_1이 속한 클래스 C_1에 대해 1보다 큰 승수를 제공한 규칙 r_1 와 집합 O_1에 속하는 규칙, 즉 C_1가 아닌 다른 클래스 C_2, C_3에 대해 1보다 작은 승수를 제공한 규칙 r_2, r_3가 보상을 받는다. $|G_1|=1$, $|O_1|=2$이므로 규칙 r_1, r_2, r_3는 각각

r_1 : $f(r_1)=\{|G_1|/(|G_1|+|O_1|)\}*(\lambda_{r1}/\Sigma_{r'\in G1}\lambda_{r'})$
$=(1/3)*(1.8/1.8)=0.33$

r_2 : $f(r_2)=\{|O_1|/(|G_1|+|O_1|)\}*\{(1/\lambda_{r2})/\Sigma_{r'\in O1}(1/\lambda_{r'})\}$
$=(2/3)*\{(1/0.8)/(1/0.8+1/0.69)\}=0.31$

r_3 : $f(r_3)=\{|O_1|/(|G_1|+|O_1|)\}*\{(1/\lambda_{r3})/\Sigma_{r'\in O1}(1/\lambda_{1'})\}$
$=(2/3)*\{(1/0.69)/(1/0.8+1/0.69)\}=0.36$

의 보상을 받는다. 사례 t_2에 대해서는 분류 시스템이 클래스 C_3를 선택하였으므로 클래스 C_2에 대해 1보다 작은 승수를 제공하는 규칙의 집합 $L_2=\varnothing$와 C_3에 대해 1보다 큰 승수를 제공하는 규칙의 집합 $G_3=\{r_3\}$ 내의 규칙들이 벌점 1을 나누어 갖게 된다. 따라서 r_1, r_2, r_3는 각각

r_1: $f(r_1)=0$

r_2: $f(r_2)=0$

r_3: $f(r_3)=-\{|G_3|/(|G_3|+|L_2|)\}*(\lambda_{r3}/\Sigma_{r'\in G3}\lambda_{r'})$
$=-1*(67.88/67.88)=-1$

의 벌점을 받게 된다. 사례 t_3에 대해서는 $G_3=\{r_3\}$, $O_3=\{r_2\}$ 이므로 r_1, r_2, r_3는 각각

r_1: $f(r_1)=0$

r_2: $f(r_2)=\{|O_3|/(|G_3|+|O_3|)\}*\{(1/\lambda_{r2})/\Sigma_{r'\in O3}(1/\lambda_{r'})\}$
$=(1/2)*\{(1/0.8)/(1/0.8)\}=0.5$

r_3: $f(r_3)=\{|G_3|/(|G_3|+|O_3|)\}*(\lambda_{r3}/\Sigma_{r'\in G3}\lambda_{r'})$
$=(1/2)*(25.89/25.89)=0.5$

의 보상을 받게 된다.

지금까지 설명한 규칙 평가 기법을 이용하여 분류 시스템에 부적합한 규칙을 좀 더 높은 확률로 선택하여 새로운 규칙으로 대체하도록 돌연 변이 연산자를 구현하였다. 돌연 변이 연산자는 규칙이 주어진 사례 집합내의 사례들을 분류하는 과정에서 얻게 되는 보상/벌점의 합계를 구한 후 그 합계액에 반비례하여 대체할 규칙을 선택한다. 예를 들면 위의 예에서 규칙 r_1, r_2, r_3이 사례 t_1, t_2, t_3를 분류하는 동안 얻게 되는 보상액은 각각 0.33, 0.81, -0.14가 된다. 각각의 합계액에 1.14를 더하여 1.47, 1.95, 1을 구한 후 돌연 변이 연산자는 규칙 r_1을 (1/1.47)/(1/1.47+1/1.95+1/1)=0.31, 규칙 r_2를 (1/1.95)/(1/1.47+1/1.95+1/1)=0.23, 규칙 r_3를 0.46의 확률로 선택하여 새로운 규칙으로 대체한다.

4.2 적합한 규칙의 생성

새로운 규칙을 생성하는 가장 간단한 방법은 규칙이 참조할 속성과 클래스, n, s, 및 is-close 규칙을 위한 a의 값을 난수 발생기를 이용하여 임의로 결정하는 것이다. 그러나 이 방법은 학습이 진행됨에 따라 규칙 집합이 필요로 하는 규칙이 아닌 임의의 규칙을 생성하게 되어 탐색의 효율성을 저하시킬 우려가 있다. 만약 규칙 집합이 어떤 클래스에 대해 낮은 분류 성능을 보인다면 규칙 집합에는 그 클래스에 속한 사례들을 구별하기 위한 적절한 규칙이 결여되어 있다고 볼 수 있다. 따라서 규칙 집합이 어떤 클래스에 대해 분류를 못하는지를 파악하여 적절한 규칙을 삽입하도록 돌연 변이 연산자를 구현하면 탐색을 좀 더 효율적으로 수행할 수 있다.

규칙 집합이 어떠한 규칙을 필요로 하는지 파악하여

적절한 규칙을 생성하기 위해 함수 miss(C_k)를 식 (2)와 같이 정의하였다:

$$miss(C_k) = \frac{\max\{mrp(C_k), mrn(C_k)\}+1}{\min\{mrp(C_k), mrn(C_k)\}+1} \quad (2)$$

식 (2)에서 mrp(C_k)는 규칙 집합이 사례 집합내의 클래스 C_k에 속한 사례를 다른 클래스로 잘못 분류한 사례의 수를 반환하는 함수이고 mrn(C_k)는 다른 클래스에 속한 사례를 C_k에 속한 것으로 분류한 사례의 수를 반환하는 함수이다.

돌연 변이 연산자는 각각의 클래스 C_k에 대해 miss(C_k)를 구한 후 그에 비례하여 규칙이 참조할 클래스로 C_k를 선택한다. 즉 돌연 변이 연산자는 miss(C_k) / $\Sigma_i$miss(C_i)의 비율로 클래스 C_k를 선택한다. 클래스를 선택한 후 mrp(C_k) ≥ mrn(C_k)이면 1보다 큰 승수를 제공하도록 규칙을 생성하고 이와 반대로 mrp(C_k) < mrn(C_k)이면 1보다 작은 승수를 제공하기 위한 규칙을 생성한다. 그 이유는 mrp(C_k)가 mrn(C_k)보다 큰 경우에는 규칙 집합이 C_k에 속한 사례를 잘 못 분류하는 경우가 다른 클래스에 속한 사례를 C_k에 속한 것으로 잘 못 분류하는 경우보다 더 많다는 의미이며 따라서 C_k에 대해 1보다 큰 승수를 제공하는 규칙을 규칙 집합에 새로 추가하면 C_k에 속한 사례에 대해 분류 성능을 향상시켜 오류를 줄일 수 있으리라는 기대 때문이다. 이와 반대로 mrn(C_k)가 mrp(C_k)보다 큰 경우에는 분류 규칙 집합은 다른 클래스에 속한 사례를 C_k에 속한 사례로 분류하는 경우가 C_k에 속한 사례를 잘 못 분류하는 경우보다 더 많은 경우이다. 이 경우에는 새로운 규칙의 생성 시에 C_k에 속하는 사례에 대해 1보다 작은 승수를 제공하는 규칙을 규칙 집합에 추가하면 C_k에 대한 가능성을 감소시켜 다른 클래스에 속한 사례를 C_k에 속한 것으로 잘 못 분류하는 경우를 감소시킬 수 있을 것이다.

규칙이 참조할 클래스와 제공하여야 할 승수의 유형이 결정되고 나면 규칙의 타입과 규칙이 참조할 속성을 난수 발생기를 이용하여 결정한 후 적절한 승수를 제공하도록 n과 s의 값을 결정한다. n과 s의 값을 결정하는 과정은 규칙의 형태에 따라 달라진다. 새로 생성해야 할 규칙이 is-high 규칙이라면 n과 s를 다음과 같이 정한다:

사례 집합내의 사례의 수를 n_{TR}라 하자.

(1) 규칙이 참조하는 속성에 대해 사례가 갖고 있는 값이 구간 [0, α], [1-α, 1]에 속하는 사례의 수 n_α와 $n_{1-\alpha}$를 구한다. 여기서 α는 $0 < \alpha < 0.5$의 값이다.

(2) i) 1보다 큰 승수를 제공하여야 할 경우

a) $n_{1-\alpha} > n_\alpha$인 경우: $n_{1-\alpha}/n_{TR}$의 확률로 $s > 1$, $0 < n < 1$이 되도록 s, n의 값을 정하고 $1- n_{1-\alpha}/n_{TR}$의 확률로 임의로 정한다 (즉, $s > 1$, $0 < n < 1$ 또는 $n > 1$, $0 < s < 1$ 이 되도록 s, n의 값을 정한다).

b) $n_{1-\alpha} < n_\alpha$인 경우: n_α/n_{TR}의 확률로 $n > 1$, $0 < s < 1$이 되도록 하고 나머지 확률로 임의로 정한다.

ii) 1보다 작은 승수를 제공하여야 할 경우

앞서 설명한 1보다 큰 승수를 제공하는 경우와는 반대로 s와 n값을 정한다.

새로 생성해야 할 규칙이 is-close 규칙인 경우에는 우선 0에서 1사이의 구간을 [0, 1/m], [1/m,2/m], ... [(1m)/m, 1]와 같이 m등분 한 후에 규칙이 참조할 속성에 대해 사례가 갖고 있는 값에 따라 각 구간에 속한 사례의 수를 구한다. 다음에 각 구간에 속한 사례의 수에 비례하여 특정 구간을 선택한 후 그 구간 내에서 난수 발생기를 이용하여 값 a를 결정한다. 선택된 구간내의 사례의 수를 n_I 사례 집합내의 사례의 수를 n_{TR}라 하면 n과 s의 값을 결정하는 과정은 다음과 같다:

(1) 1보다 작은 승수를 제공하여야 할 경우:

n_I/n_{TR}의 확률로 $n > 1$, $0 < s < 1$이 되도록 n과 s의 값을 결정하고 $1- n_I/n_{TR}$의 확률로 임의로 정한다.

(2) 1보다 큰 승수를 제공하여야 할 경우:

n_I/n_{TR}의 확률로 $s > 1$, $0 < n < 1$이 되도록 s, n의 값을 결정하고 나머지 확률로 임의로 정한다.

이제까지 설명한 방법으로 규칙을 생성하도록 돌연 변이 연산자를 구현하였다.

5. 학습 시스템의 성능 평가

다양한 사례 집합을 이용하여 학습 시스템의 학습 능력을 평가 하였다. 학습 시스템의 성능 평가를 위해 사용한 사례 집합은 다음과 같다.[2)]

• 붓꽃 사례 집합: 3가지 종류의 붓꽃으로부터 얻어진 150개의 사례 집합으로 각각의 사례는 꽃잎의 길이, 꽃잎의 넓이, 꽃받침의 길이, 꽃받침의 넓이의 4가지 속성 값과 그 사례가 속한 붓꽃의 종류를 나타내는 값으로 표현된다.

• 유리 사례 집합: 6 종류의 유리 파편들로부터 얻어진 214개의 사례로 구성되어 있으며 각각의 사례는 유리를 구성하는 9가지 물질의 성분비 및 그 사례가 속한 유

2) 이들 사례 집합은 UCI machine learning repository에 있는 사례 집합들로 ftp를 이용하여 ics.usi.edu의 pub/machine-learning-databases 디렉토리에서 습득하였음.

리의 종류를 나타내는 값으로 표현된 사례 집합이다.

- 레이더 시그널 사례 집합: 올바른 경우와 잘못된 경우의 351개 레이더 시그널 사례가 34가지의 속성에 대한 속성 값과 그 사례가 속한 클래스로 표현 된 사례 집합이다.
- 콩의 질병 사례 집합: 콩에 감염될 수 있는 15가지의 질병으로부터 얻어진 사례 집합으로 290개의 질병 사례가 35개의 속성 값으로 표현되어 있다.
- 당뇨 환자 사례 집합: 당뇨 환자인 경우와 정상인인 경우로 분류되는 768개의 사례로 구성되었으며, 각각의 사례는 8가지 속성 값과 그 사례가 속한 클래스로 표현된다.

사례 집합을 크기가 같은 두 개의 부분 집합인 훈련 사례 집합과 평가 사례 집합으로 나누어 훈련 사례 집합을 이용하여 규칙 집합을 습득한 후 평가 사례 집합으로 습득된 규칙 집합의 분류 성능을 평가하는 과정을 각각의 사례 집합에 대해 20회 씩 반복하였다. 유전자 알고리즘에 대해 사용한 매개 변수의 값은 다음과 같다.

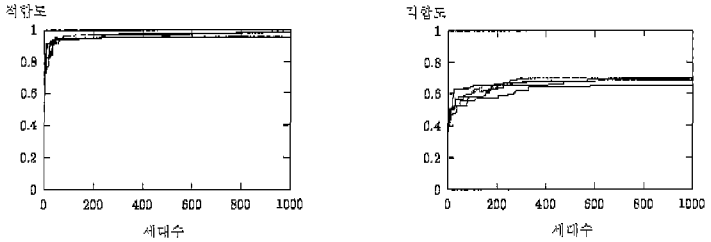
- 개체 집단의 크기: 15
- 규칙 집합내의 규칙 수의 상한: 50
- 규칙 집합내의 규칙 수의 하한: 3
- 교배 연산자 비율: 98%
- 역치 연산자 비율: 5%
- 돌연변이 연산자 비율: 40%

위에서 언급한 매개 변수 중 규칙 집합내의 규칙 수의 상한과 하한은 학습 과정의 초기 단계에서 개체 집단의 생성시 규칙 집합의 크기를 제한 하기 위한 것이며 학습이 진행되는 과정에서는 규칙 집합의 크기에 제한을 두지 않았다. 따라서 학습 과정에서 규칙 집합은 임의의 수의 규칙을 갖을 수 있으며 이 경우에 학습 시스템이 학습 과정을 통해 필요한 적정 수의 규칙으로 이루어진 규칙 집합을 습득할 수 있도록 하기 위해서는 규칙 집합의 크기가 증가하는 것을 억제하기 위한 수단이 필요하다. 이를 위한 수단으로 적합도 함수가 이용되었다. 학습 시스템에서 임의의 규칙 집합 RS의 적합도는 다음과 같이 평가된다.

$$fitness(RS) = \frac{n_c}{n_{TR}} - \frac{|RS|}{(m \times n \times 50)} \qquad (3)$$

식 (3)에서 n_c는 훈련 사례 집합내의 사례 중 규칙 집합이 올바르게 분류한 사례의 수, n_{TR}은 훈련 사례 집합내의 사례의 수를 나타내고 |RS|, m, n은 각각 규칙 집합내의 규칙의 수, 훈련 사례 집합내의 클래수의 수, 훈련 사례 집합내의 속성의 수를 나타낸다. 식 (3)의 적합도 함수는 기본적으로 규칙 집합의 적합도를 주어진 사례를 얼마나 올바르게 분류하는가 하는 분류의 정확도로 평가하지만 이를 규칙 집합의 크기에 비례하여 감소시킴으로써 새로운 개체 집단의 생성 과정에서 적은 수의 규칙을 갖는 규칙 집합이 더 유리하도록 하여 개체 집단내의 규칙 집합의 크기가 증가하는 것을 억제한다. 적절한 학습 환경의 구축을 위한 다양한 실험에서 식 (3)의 적합도 함수는 규칙 집합의 크기를 적절하게 유지하도록 하는 것으로 나타났으며 그 밖의 다른 매개 변수가 학습 능력에 미치는 영향을 평가한 결과에서는 개체 집단의 크기는 학습 능력의 향상에 크게 영향을 미치지 않는 것으로 나타났으나 돌연 변이 연산자의 비율은 시스템의 성능에 영향을 많이 미치는 것으로 나타났으며 역치 연산자는 교배 연산자나 돌연 변이 연산자에 비해 상대적으로 큰 영향을 미치지는 않는 것으로 밝혀 졌다.

학습 시스템은 규칙 집합의 습득 시 100세대동안 개체 집단의 평균 적합도의 증가량이 0.0005이하 이거나 최대 적합도의 증가량이 0.001 이하이면 학습 과정을 종료하도록 하였으며, 이 경우 규칙 집합의 습득에 소요되는 시간은 평균적으로 붓꽃 사례 집합에 대해서는 400세대, 당뇨 환자의 사례 집합에 대해서는 500세대, 유리 사례 집합에 대해서는 800세대, 레이더 시그널 사례 집합과 콩의 질병 사례 집합에 대해서는 각각 1200세대와 1500세대 정도의 시간이 필요하였다. 그림 6은 붓꽃 사례 집합과 유리 사례 집합에 대한 5회의 학습 과정에서



(a) 붓꽃 사례 집합 (b) 유리 사례 집합

그림 6 규칙 집합의 습득 과정

매 세대별 개체 집단의 최상의 적합도를 보인 것이다.

학습 시스템의 각각의 사례 집합에 대한 학습 결과를 표 2에 정리하였다. 표 2는 학습 시스템이 붓꽃 사례 집합에 대해 평균적으로 98.3%의 정확도를 갖는 규칙 집합을 습득하였으며 습득된 규칙 집합을 이용하여 구축된 분류 시스템은 주어진 새로운 사례를 94.2%의 정확도를 가지고 분류하였음을 보인다.

표 2 학습 시스템의 성능 평가 결과

사례집합	분류의 정확도		세대수
	훈련사례	평가사례	
붓 꽃	98.3±1.2	94.2±1.4	400
유 리	68.6±3.6	58.0±2.5	800
당 뇨	78.3±1.5	74.9±1.3	500
헤이더	93.7±2.3	86.6±2.7	1200
콩의질병	61.9±4.3	52.7±3.5	1500

6. 다중 규칙 집합을 이용한 분류 시스템의 구현

학습 과정의 초기 단계에서 학습 시스템은 난수 발생기를 이용하여 일정 수의 규칙 집합으로 구성된 개체 집단을 생성한다. 계속되는 학습 과정에서는 개체 집단 내의 구성원에 유전 연산자들을 적용하여 새로운 개체 집단을 생성하는 과정을 만족할 만한 규칙 집합을 습득할 때 까지 반복하는데 이 과정에서 유전 연산자를 적용하기 위한 규칙 집합의 선택, 임의의 규칙을 다른 새로운 규칙으로 교체하기 위한 규칙의 선택 및 규칙의 생성 등에 난수 발생기가 이용된다. 난수 발생기에 의존한 학습 과정은 난수 발생기에 사용되는 초기 값이 변함에 따라 다른 탐색 공간을 탐색하여 결과적으로 다른 규칙 집합을 습득하게 된다. 이러한 특성을 이용하여 주어진 사례 집합에 대해 학습 시스템을 반복 실행하여 서로 다른 다수의 규칙 집합을 습득한 후 이들을 이용하여 분류 시스템을 구축하기 위한 기법을 개발하고 그 성능을 평가하였다.

6.1 다중 규칙 집합을 이용한 분류 시스템에서 의사 결정 기법

다수의 규칙 집합을 이용하여 분류 시스템을 구축하기 위해서는 이들 규칙 집합들이 제공하는 분류 결과를 취합하여 최종 결론을 도출해 내는 기법이 필요하다. 이를 위해 이용 가능한 기법 중 하나는 단순 보우팅(voting)을 이용하는 것이다. 이 기법하에서 분류 시스템 내의 규칙 집합은 2장에서 논한 방법에 따라 사후 가능성이 가장 큰 클래스를 정하여 그 결과를 분류 시스템에 제공하고 분류 시스템은 다수결의 원칙에 따라 주어진 사례가 속한 클래스를 최종적으로 선택한다. 이러한 단순 보우팅 기법은 구현이 간단한 반면에 사례가 다른 클래스에 속할 사후 가능성을 의사 결정과정에 반영하지 못하는 단점이 있다.

다수의 규칙 집합을 이용한 분류 시스템의 구현 시 단순 보우팅 기법보다는 좀 더 정확한 분류 결과를 얻을 수 있으리라는 기대하에 규칙 집합이 제공하는 정규화된 사후 가능성을 취합하여 그 결과에 따라 사례가 속할 클래스를 선택하는 기법을 개발하였다. 이 기법하에서 규칙 집합은 클래스에 대한 사후 가능성을 식 (4)에 따라 정규화 하여 그 결과를 분류 시스템에 제공한다.

$$NO(C_j') = \frac{O(C_j')}{\max_k\{O(C_k')\}} \quad (4)$$

식 (4)에서 $NO(C_j')$, $O(C_j')$는 각각 클래스 C_j에 대한 정규화 된 사후 가능성 및 사후 가능성을 나타내고 $max_k\{O(C_k')\}$는 클래스들에 대한 사후 가능성 중 가장 큰 값을 반환하는 함수이다. 분류 시스템은 정규화 된 사후 가능성들을 각각의 클래스 C_j에 대해 식 (5)에 따라 취합한 후 CE값이 가장 큰 클래스를 주어진 사례가 속한 클래스로 선택한다.

$$CE(C_j') = \left(\prod_{i=1}^{n}(1+NO_{RS_i}(C_j'))\right)^{\frac{1}{n}} \quad (5)$$

식 (5)에서 $NO_{RSi}(C_j')$는 분류 시스템 내의 규칙 집합 RS_i가 클래스 C_j에 대해 제공하는 정규화 된 사후 가능성을, n은 분류 시스템내의 분류 규칙 집합의 수를 나타낸다.

식 (4)를 이용한 정규화 과정은 한 규칙 집합의 사후 가능성이 다른 규칙 집합의 사후 가능성에 비해 지나치게 큰 경우로 인해 의사 결정 과정에 절대적인 영향을 미치지 않도록 규칙 집합들 사이에 존재하는 사후 가능성에 대한 편차를 줄이기 위한 조치이다. 식 (5)에서는 0에 근접한 정규화 된 사후 가능성이 전체 의사 결정에 영향을 미치지 않도록 하기 위해 정규화 된 사후 가능성에 1의 값을 더하였다.

6.2 다중 규칙 집합을 이용한 분류 시스템의 성능 평가

다수의 규칙 집합을 이용하여 분류 시스템을 구축하는 한가지 방법은 미리 정해진 수 만큼의 규칙 집합을 습득한 후 이를 이용하여 분류 시스템을 구축하는 것이다. 그러나 이 방법보다 좀 더 효율적인 방법은 다수의 규칙 집합에서 분류 시스템의 성능을 극대화 할 수 있는 규칙 집합의 조합을 찾아 이를 이용하여 분류 시스템을 구축하는 것이다.

5장에서 기술한 사례 집합을 이용하여 다중 규칙 집합을 이용한 분류 시스템의 성능을 다음과 같이 평가하였다. 우선 사례 집합을 크기가 같은 두 개의 부분 집합인 훈련 사례 집합과 평가 사례 집합으로 나눈 후 훈련 사례 집합에 대해 학습 알고리즘을 반복 실행하여 20개의 규칙 집합을 구한다. 그런 다음 습득된 20개의 규칙 집합으로부터 유전자 알고리즘을 이용하여 최적의 규칙 집합의 조합을 찾아내어 이를 이용하여 분류 시스템을 구축 하고 평가 사례를 이용하여 분류 시스템의 성능을 평가한다. 이와 같은 실험을 각각의 사례 집합에 대해 5회씩 반복하였다.

규칙 집합의 조합을 찾기 위한 유전자 알고리즘에서 개체 집단의 구성원은 습득한 규칙 집합의 수 만큼의 이진 문자로 표현되며 각각의 이진 문자는 상응하는 규칙 집합이 구성원에 포함되는지의 여부에 따라 0과 1로 표현된다. 즉 개체 집단의 각각의 구성원은 습득된 규칙 집합들 중의 일부로 구성된다. 규칙 집합의 조합 탐색을 위한 초기 단계에서는 일정 수의 구성원을 갖는 개체 집단을 난수 발생기를 이용하여 생성한 후 계속되는 진화 과정에서는 개체 집단내의 구성원에 교배 연산자, 돌연변이 연산자를 적용하여 새로운 개체 집단을 생성하는 과정을 만족할 만한 규칙 집합의 조합을 습득할 때까지 반복한다. 교배 연산자는 현재의 개체 집단에서 2개의 구성원을 선택하여 구성원내의 규칙 집합의 일부를 교환하여 새로운 개체를 생성한다. 돌연 변이 연산자는 선택된 구성원 내의 이진 문자 하나를 다른 문자로 교체하여 규칙 집합을 제거하거나 추가하는 과정을 통해 새로운 개체를 생성한다. 각각의 구성원의 적합도는 구성원 내의 규칙 집합들로 분류 시스템을 구축한 후 주어진 훈련 사례 집합을 어느 정도 정확하게 분류해 내느냐 하는 분류의 정확도로 평가 한다.

단일 규칙 집합을 이용하여 구축한 분류 시스템과 다수의 규칙 집합을 이용하여 구축한 분류 시스템의 성능을 표 3에 비교하였다. 표 3에서 '단일'은 하나의 규칙 집합을 이용하여 분류 시스템을 구축한 결과를 '다중'은 다수의 규칙 집합을 이용하여 구축한 분류 시스템의 성능을 평가한 결과이다.

표 3은 단일 규칙 집합을 이용한 분류 시스템이 붓꽃 사례 집합에 대한 실험에서 평균적으로 훈련 사례 집합을 98.3%, 평가 사례 집합내의 사례들의 94.2%를 올바르게 분류한 반면에 다중 규칙 집합을 이용한 분류 시스템은 훈련 사례 집합을 100.0%, 평가 사례 집합내의 사례들의 96.0%를 올바르게 분류 하였음을 보인다. 또한 다중 규칙 집합을 이용한 경우에 분류 시스템의 성능이 평균적으로 6.4% 향상되었음을 보인다.

표 3 다중 규칙 집합을 이용한 분류 시스템의 성능 평가 결과

사례 집합	훈련사례집합		평가사례집합	
	단일	다중	단일	다중
붓 꽃	98.3	100.0	94.2	96.0
유 리	68.6	80.4	58.0	65.8
당 뇨	78.3	82.0	74.9	76.3
레이더	93.7	98.2	86.6	92.3
콩의질병	61.9	76.1	52.7	68.2
평균	80.2	87.3	73.3	79.7

본 연구에서 구축한 학습 시스템을 이용하여 얻어진 분류 시스템의 성능을 신경망과 결정 트리 알고리즘을 이용하여 구축된 분류 시스템의 성능과 비교하였다. 표 4에서 '단일'과 '다중'은 단일 규칙 집합과 다수의 규칙 집합을 이용하여 구축한 분류 시스템을 나타내고 '신경망'과 'C4.5'는 각각 신경망과 C4.5 결정 트리 알고리즘을 이용하여 구축한 분류 시스템의 성능을 보인 것이다. 신경망과 C4.5에 대한 성능은 주어진 사례 집합의 70%를 훈련 사례로 나머지 30%를 평가 사례로 하여 5회의 실험을 시행하여 그 결과를 평균한 것이다[14]. 반면에 '단일'과 '다중'에 대한 결과는 사례 집합의 50%를 훈련 사례로 나머지 50%를 평가 사례로 하여 평가한 결과이다.

표 4 학습 시스템의 성능 비교

사례 집합	단일	다중	신경망	C4.5
붓 꽃	94.2	96.0	95.7	92.7
당 뇨	74.9	76.3	67.7	71.4

표 4의 결과는 본 연구를 통해 구축한 학습 시스템이 신경망과 결정 트리 알고리즘에 견줄 만 한 학습 능력이 있음을 보인다. 붓꽃 및 당뇨 사례 집합에 대해 단일 규칙 집합을 이용한 분류 시스템은 신경망과 결정 트리 알고리즘에 상응하는 분류 성능을 보였으며 다중 규칙 집합을 이용한 분류 시스템은 신경망과 결정 트리 알고리즘을 이용하여 구축한 분류 시스템의 성능보다 더 나은 분류 성능을 보였다.

7. 결 론

주어진 사례의 집합으로부터 사례들을 분류할 수 있는 PROSPECTOR 규칙 유형의 분류 규칙을 습득하기 위한 귀납적 학습 환경을 유전자 알고리즘을 이용하여

구현하였다. 유전자 알고리즘을 이용한 학습 시스템의 구현에서 개체 집단은 일정 수의 규칙 집합으로 구성되고 규칙 집합의 적합도는 사례 집합에 대한 분류의 정확도로 평가한다. 학습 과정에서는 현재의 개체 집단에 유전 연산자를 적용하여 새로운 개체 집단을 생성하는 과정을 만족할 만한 규칙 집합을 습득할 때까지 반복한다. 유전자 알고리즘으로 하여금 탐색을 좀 더 효율적으로 수행할 수 있도록 하기 위해 분류 규칙의 성능을 평가하기 위한 기법과 분류 작업에 적합한 규칙을 생성하기 위한 기법에 대해 연구하였다. 분류 시스템의 성능 향상을 위해 다수의 규칙 집합을 이용하여 분류 시스템을 구축하기 위한 기법을 개발하고 다양한 사례 집합을 이용하여 학습 시스템의 학습 능력을 평가 하였다.

구축된 학습 시스템은 다양한 사례 집합에 대해 신경망, 결정 트리 등의 다른 학습 알고리즘과 견줄 만 한 학습 능력을 보였다. 따라서 본 연구의 결과로 구축된 학습 시스템은 신경망, 결정 트리 등 기존의 학습 알고리즘의 대안으로 여러 응용 분야에서 활용될 수 있으리라 사료된다. 또한 다중 규칙 집합을 이용한 분류 시스템의 구축에 관한 연구는 유전학 기반 학습 시스템 및 신경망, 결정 트리 등을 이용한 학습 시스템의 성능 향상을 위한 일반적인 기법으로 활용 가능하리라 사료된다.

향후 연구 과제로는 논리 연산자의 도입을 통한 규칙 구문 형태의 확장 및 규칙의 표현 능력 강화, 규칙의 성능 평가 기법 및 규칙 생성 기법의 개선, 탐색의 지역화를 위한 돌연변이 연산자의 도입, 학습 진행 과정을 보여줄 수 있는 시각화 기법의 개발 등 보다 나은 학습 환경의 구축을 위한 연구가 필요하겠다.

참 고 문 헌

[1] D. E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning," Addison-Wesley, 1989.

[2] M. Srinivas and L. M. Parnaik, "Genetic algorithms: a survey," *IEEE Computer*, Vol. 27, pp. 17-26, June 1994.

[3] R. Duda, P. Hart and J. Nilsson, "Subjective Bayesian methods for rule-based inference systems," in *Proc. National Computer Conference*, pp. 1075 1082, 1976.

[4] K. De Jong, "Genetic algorithm-based learning," *Machine Learning:An Artificial Intelligence Approach*, Vol. 3, Y. Kodratoff and R. S. Michalski, Eds. San Mateo, CA:Morgan Kaufmann, pp. 611-638, 1990.

[5] G. Roberts, "Dynamic planning for classifier systems," in *Proc. 5th Int. Conf. Genetic Algorithms*, pp. 231-237, 1993.

[6] S. Wilson and D. Goldberg, "A critical review of classifier systems," in *Proc. 3rd Int. Conf. Genetic Algorithms*, pp. 244-255, 1989.

[7] J. Oliver, "Discovering individual decision rules: an application of genetic algorithms," in *Proc. 5th Int. Conf. Genetic Algorithms*, pp. 216-222, 1993.

[8] M. G. Cooper and J. J. Vidal, "Genetic design of fuzzy controllers: the cart and jointed-pole problem," *The Third Int. Conf. Fuzzy Systems*, Vol.3, pp. 1332-1337, 1994.

[9] A. Homaifar and E. McCormick, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms," *IEEE Trans. Fuzzy Systems*, Vol. 3, No. 2, pp. 129-139, 1995.

[10] C. K. Chiang, H. Y. Chung, and J. J. Lin, "A self-learning fuzzy logic controller using genetic algorithms with reinforcements," *IEEE Trans. Fuzzy Systems*, Vol. 5, pp. 460-467, 1997.

[11] Y. Yuan, and H. Zhaung, "A genetic algorithm for generating fuzzy classification rules," *Fuzzy Sets Syst.*, Vol. 84, No. 1, pp. 1-19, Nov. 1996.

[12] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, "Selecting fuzzy if-then rules for classification problems using genetic algorithms," *IEEE Trans. Fuzzy Systems*, Vol.3, pp. 250-270, 1995.

[13] H. Ishibuchi and T. Nakashima, "Improving the Performance of Fuzzy Classifier Systems for Pattern Classification Problems with Continuous Attributes," *IEEE Transactions on industrial electronics*. Vol. 46, No. 6, December 1999, pp. 1057-1068.

[14] O. Cordon, M. Jesus, and F. Herrera, "Genetic Learning of Fuzzy Rule-Based Classification Systems Cooperating with Fuzzy Reasoning Methods," *Int. Journal of Intelligent Systems*, Vol. 13, pp. 1025-1053, 1998.

김 영 준

1984년 고려대학교 산업공학과 졸업(학사). 1990년 미국 Univ. of Houston 전산학(석사). 1996년 미국 Univ. of Houston 전산학(박사). 1997년 ~ 현재 상명대학교 정보통신학부 조교수. 관심분야는 분산처리, 기계학습, 전문가 시스템