군집화와 유전 알고리즘을 이용한 거친-섬세한 분류기 앙상블 선택

(Coarse-to-fine Classifier Ensemble Selection using Clustering and Genetic Algorithms)

김 영 원 † 오 일 석 ††
(Young-Won Kim) (Il-Seok Oh)

요 약 좋은 분류기 앙상블은 분류기 간에 상호 보완성을 갖추어 높은 인식 성능을 보여야 하며, 크기가 작아 계산 효율이 좋아야 한다. 이 논문은 이러한 목적을 달성하기 위한 거친-섬세한 (coarse-to-fine) 단계를 밟는 분류기 앙상블 선택 방법을 제안한다. 이 방법이 성공하기 위해서는 초기 분류기 풀 (pool)이 충분히 다양해야 한다. 이 논문에서는 여러 개의 서로 다른 분류 알고리즘과 아주 많은 수의 특징 부분집합을 결합하여 충분히 큰 분류기 풀을 생성한다. 거친 선택 단계에서는 분류기 풀의 크기를 적절하게 줄이는 것이 목적이다. 분류기 군집화 알고리즘을 사용하여 다양성을 최소로 희생하는 조건하에 분류기 풀의 크기를 줄인다. 섬세한 선택에서는 유전 알고리즘을 이용하여 최적의 앙상블을 찾는다. 또한 탐색 성능이 개선된 혼합 유전 알고리즘을 제안한다. 널리 사용되는 필기 숫자 데이타베이스를 이용하여 기존의 단일 단계 방법과 제안한 두 단계 방법의 성능을 비교한 결과 제안한 알고리즘이 우수함을 입증하였다.

키워드 : 분류기 앙상블 선택, 유전 알고리즘, 분류기 다양성, 분류기 군집화, 필기 숫자 인식

Abstract The good classifier ensemble should have a high complementarity among classifiers in order to produce a high recognition rate and its size is small in order to be efficient. This paper proposes a classifier ensemble selection algorithm with coarse-to-fine stages. For the algorithm to be successful, the original classifier pool should be sufficiently diverse. This paper produces a large classifier pool by combining several different classification algorithms and lots of feature subsets. The aim of the coarse selection is to reduce the size of classifier pool with little sacrifice of recognition performance. The fine selection finds near-optimal ensemble using genetic algorithms. A hybrid genetic algorithm with improved searching capability is also proposed. The experimentation uses the worldwide handwritten numeral databases. The results showed that the proposed algorithm is superior to the conventional ones.

Key words : Classifier Ensemble Selection, Genetic Algorithm, Classifier Diversity, Classifier Clustering, Handwritten Numeral Classification

1. 서 론

패턴인식 시스템의 인식 성능을 높이기 위한 다중 분류기 결합(Multiple Classifier Combination) 아이디어는 최근에도 아주 활발히 연구가 진행 중이다[Ueda00, Zhou02, Kittler03, Granitto05, Pedrajas05, Fumera05, Wanas06, Rodriguez06, Banfield07, Sohn07].

† 정 회 원 : 한국 전자통신연구원 우정기술연구센터 자동구분처리연구팀 연구원
everywkim@etri.re.kr
†† 종신회원 : 전북대학교 전자정보공학부 교수
isoh@chonbuk.ac.kr
논문접수 : 2007년 1월 22일
심사완료 : 2007년 6월 13일

Ho는 다중 분류기 시스템 설계를 범위 최적화(coverage optimization)와 결정 최적화(decision optimization) 문제로 구분하였다[Ho02]. 결정 최적화 문제는 다중 분류기의 출력을 어떻게 결합하여야 최적의 성능을 얻을 수 있는지를 다룬다. 다중 분류기 연구의 초창기에는 이 문제의 해결에 연구가 집중되었으며 현재까지 아주 많은 연구 결과가 발표되어 있다[Ueda00, Kittler03, Fumera05]. 범위 최적화는 분류기 결합에 참여할 분류기 집합, 즉 분류기 앙상블을 결정하는 문제이다. 아무리 결합 방법이 좋더라도 참여하는 분류기들의 상호 보완성이 약하다면 의미 있는 성능 향상을 기대하기 어렵다. 따라서 범위 최적화는 결정 최적화에 못지않게 중요하다.

최근에는 범위 최적화 문제에 연구가 집중되고 있는 경향을 보이고 있다[Zhou02, Granitto05, Pedrajas05, Wanas06, Rodriguez06, Banfield07, Sohn07]. 범위 최적화 문제를 해결하는 접근 방법에는 훈련 샘플 집합을 여러 개의 부분집합으로 나누어 다중 분류기를 만드는 bagging이나 boosting같은 접근 방법[Theodoridis06], 특징 집합을 여러 개의 부분집합으로 나누어 다중 분류기를 만드는 방법[Rodriguez06], 분류기 훈련 과정에서 분류기 간의 상호 작용을 고려하는 적응적인 방법[Pedrajas05, Wanas06], 많은 분류기를 만들어 놓고 이들 중에서 일부를 선택하는 과생산 후 선택(overproduction and selection) 방법[Zhou02] 등이 있다. 또한 이들 방법을 실험적으로 비교 분석한 논문들이 발표되었다[Granitto05, Banfield07, Sohn07].

과생산 후 선택 방법을 구현하기 위해서는 분류기 앙상블 선택 문제를 해결해야 한다. 이 문제는 D 개의 분류기를 갖는 분류기 풀(*classifier pool*)에서 d 개의 분류기를 갖는 분류기 부분집합을 선택하는 것으로 정의할 수 있다. 이렇게 선택된 분류기 부분집합을 *분류기 앙상블*(*classifier ensemble*)이라 부른다. 탐색 공간의 크기가 ${}_DC_d$ 이므로 모든 후보 해를 탐색해보는 방법은 최적의 해를 보장하나 D가 커질수록 시간비용이 지수적으로 커지는 문제를 안고 있다. 따라서 주어진 계산 시간 내에 높은 품질의 부최적해(suboptimal solution)를 찾는 알고리즘이 필요하다.

분류기 앙상블 선택 알고리즘을 설계하는데 중요하게 고려해야할 사항을 아래와 같이 나열할 수 있다.

- 충분히 큰 분류기 풀을 다룰 수 있어야 한다. 작은 분류기 풀은 충분히 다양한 분류기들을 담을 수 없기 때문에, 거기에서 찾은 분류기 앙상블은 상호보완성 측면에서 한계를 가질 수밖에 없다.
- 적절한 계산 시간 안에 분류기 앙상블을 찾아야 한다. 분류기 풀의 크기에 따라 탐색 공간이 지수적으로 커지므로, 계산 효율을 중요하게 고려해야만 한다.
- 분류기 앙상블은 상호 보완성이 좋아 높은 인식 성능을 보장해야 한다. 인식 성능이라는 기준 함수에 따라 최적에 근접한 해를 찾아야만 한다.
- 분류기 앙상블은 작을수록 좋다. 인식에 걸리는 시간은 분류기 앙상블의 크기에 비례한다.

Partridge는 분류기들을 개별적으로 평가한 후 가장 좋은 분류기만을 선택하여 사용하는 단순한 방법을 제안하였다[Partridge96]. Sharkey는 크기가 45인 분류기 풀에서 크기 3인 분류기 앙상블을 찾는데, ${}_{45}C_3$개의 후보 중에 100개를 임의 선택하고 그들 중에 가장 좋은 해를 선택하였다[Sharkey00]. 이러한 방법으로 앙상블 선택의 필요성을 강조하였다. Giacinto는 분류기가 각 샘플에 대해 옳게 인식했는지 여부를 0과 1로 표현한 인식 결과 벡터를 만들어, 분류기를 인식 결과 벡터로 표현하였다. 이 분류기들을 군집화하고, 각 군집에서 대표 분류기를 추출하여 그들로 다중 결합기를 구성하였다[Giacinto01]. 이 방법은 분류기의 인식 성능이 높은 경우 인식 결과 벡터가 서로 매우 유사하여 군집화에 어려움이 발생하는 한계가 있다. Hao는 분류기 앙상블 선택 문제와 특징 선택 문제의 유사성을 언급하고 특징 선택에서 개발된 순차 탐색 알고리즘을 도입하였다[Hao03]. 이 방법은 greedy한 탐색 방법이므로 국소 최적점에 빠지는 문제를 지니고 있다. 분류기 앙상블 선택은 탐색 문제이므로 유전 알고리즘을 사용하여 해결할 수 있다. Zhou는 각 분류기가 실수 가중치를 가지도록 염색체를 표현하였으며, 유전 알고리즘이 출력하는 염색체에서 가중치가 임계치보다 큰 분류기를 선택한다[Zhou02]. 이 논문에서는 크기가 20인 분류기 풀에서 실험하였다. 따라서 충분히 다양한 분류기를 다루었다고 할 수 없다.

우리 논문은 분류기 앙상블 선택을 위한 두 단계 알고리즘을 제안한다. 기본적인 전략은 아주 많은 분류기를 만들어 충분히 다양한 초기 분류기 풀을 만들고, 효과적인 탐색 알고리즘을 이용하여 최적의 분류기 앙상블을 찾는 것이다. 우선 크기가 수백 또는 수천에 가까운 분류기 풀을 만든다. 첫 단계에서는 다양성 희생을 최대한 적게 하는 조건하에 풀의 크기를 백 정도로 줄이는 거친 선택을 수행한다. 두 번째 단계에서는 섬세한 탐색을 수행하여 십 정도의 크기를 가진 분류기 앙상블을 만든다.

이 아이디어를 구현하는데 있어 가장 중요하게 고려해야할 사항은 분류기의 다양성(diversity)이다. 아주 높은 성능을 갖지만 서로 비슷한 분류기들을 결합 할 때에 비해 비록 성능이 낮은 분류기를 포함하더라도 서로 인식 성향이 다른 분류기들을 결합할 때가 더 높은 성능을 보장한다. 따라서 선택 과정 곳곳에서 다양성을 확보할 수 있도록 알고리즘을 설계하는 것이 중요하다.

우리 방법은 초기 분류기 풀이 충분히 많고 충분히 다양한 분류기를 갖도록 여러 분류 알고리즘을 사용한다. 신경망, SVM, KNN 등과 같이 서로 다른 분류 알고리즘을 사용하여 다양성을 얻는다. 또한 여러 특징 추출 알고리즘을 사용하여 다양성을 추가로 확보하고 특징 부분집합을 생성하여 충분히 많은 분류기를 생성하였다.

첫 단계를 위해서는 군집화 방법을 사용하였다. 서로 다른 군집에 속한 분류기들은 서로 다른 인식 경향을 가지므로, 각 군집에서 대표 분류기를 추출하면 다양성을 유지하면서 분류기 풀을 크기를 줄일 수 있다. 군집

화 알고리즘에서 군집 개수를 설정할 수 있으므로 원하는 크기로 쉽게 줄일 수 있다.

두 번째 단계를 위해서는 유전 알고리즘을 사용한다. 단순 유전 알고리즘을 제시한 후, 탐색 성능 개선을 목적으로 혼합 유전 알고리즘을 제시한다. 이 혼합 알고리즘은 특징 추출을 위해 고안된 것을 도입하여 사용 한다[Oh04].

필기 숫자 인식을 실험 대상으로 하였다. 실험용 데이타베이스는 세계적으로 표준적으로 사용하는 CENPARMI 데이타베이스와 NIST 데이타베이스를 사용하였다. 기존 알고리즘과 인식 성능과 다양성의 두 가지 측면에서 비교 분석하였다. 실험 결과 이 논문에서 제안한 방법이 기존 방법에 비해 우수하였으며, 혼합 유전 알고리즘을 사용하면 추가적인 성능 향상을 얻을 수 있음을 보였다. 제안한 알고리즘이 찾은 최적의 분류기 앙상블의 크기는 10정도였다.

2장에서는 두 단계 알고리즘의 전체 과정을 설명하고, 초기 분류기 풀의 생성 방법을 기술한다. 3장은 군집화에 의한 거친 선택 단계를 설명한다. 4장에서는 유전 알고리즘에 의한 분류기 앙상블 선택 방법을 설명한다. 먼저 단순 유전 알고리즘을 설명하고, 지역 탐색 연산을 추가한 혼합 유전 알고리즘을 설명한다. 5장에서는 기존 알고리즘과 제안한 알고리즘의 성능을 비교한다. 6장에서는 결론과 향후 연구에 대하여 기술한다.

2. 두 단계 분류기 앙상블 선택

2.1 전체 처리 과정

그림 1은 두 단계 접근 방법을 보이고 있다. 초기 분류기 풀의 크기는 수백 또는 수천이 가능하다. 5장의 실험에서는 450개를 갖도록 하였다. 거친 선택 단계에서 사용하는 군집화 알고리즘은 큰 분류기 풀을 다룰 수 있을 만큼 충분히 빠르게 작동한다. 또한 군집별로 대표 분류기를 추출하므로 다양성을 유지한다. 섬세한 선택을 위해서는 유전 알고리즘을 사용한다. 실험에서는 크기가 90인 분류기 풀에서 탐색을 수행한다. 필기 숫자 인식 문제에서 크기가 10정도인 분류기 앙상블을 가장 좋은 해로 찾아냈다.

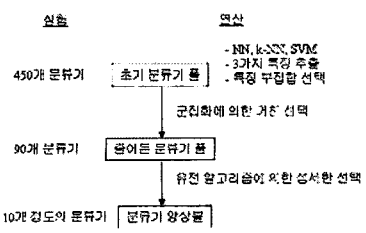

그림 1 제안한 방법의 전체 처리 과정

2.2 초기 분류기 풀 생성

이 단계의 목표는 충분히 다양한 분류기를 확보하는 것이다. 이 논문은 서로 다른 분류 알고리즘과 서로 다른 특징 추출 알고리즘을 사용함으로써 방대한 양의 분류기를 생성하여 이 목적을 달성한다. 이 논문에서는 MLP, KNN, SVM이라는 세 종류의 분류 알고리즘을 사용하였다. 또한 서로 다른 특징 추출 알고리즘을 통해 다양성을 확보한다. 우리 실험에서는 필기 숫자 인식 문제를 다루었으며 주어진 비트맵에서 세 가지 종류의 특징을 추출하여 다양성을 높였다. 또한 세 종류의 특징 벡터 각각으로부터 임의 선택에 의해서 특징 부분집합 50개를 만들어 사용하였다. 따라서 총 3*3*50=450개의 분류기를 만들어 분류기 풀을 구성하였다. 주어진 패턴이 특징 벡터일 경우에는 특징 부분집합 선택으로 분류기의 다양성을 높일 수 있다.

3. 군집화에 의한 거친 선택

여기서는 분류기들을 인식 성향에 따라 군집화하는 과정을 설명한다. 하나의 분류기 c_i는 아래와 같은 인식 결과 벡터로 표현한다.

$$c_i = (t_1, t_2, ..., t_N)$$

N은 입력 패턴의 개수이고, t_k는 k 번째 입력 패턴이 옳게 인식되었는지 여부를 나타내는 변수이다. t_k는 0 또는 1 값을 가지며, 0은 오인식, 1은 정인식을 뜻한다.

3.1 분류기 사이의 거리 측정

분류기들의 군집화를 위해 분류기들 간의 거리를 측정해야한다. 이 논문에서는 CD(Compound Diversity)를 사용하였다[Giacinto01]. $prob(c_i\ fails,\ c_j\ fails)$는 두 개의 분류기 c_i와 c_j가 동시에 틀리는 확률을 의미한다. N^{11}은 분류기 c_i와 c_k 둘 다 맞추는 샘플의 수, N^{00}는 둘 다 틀리는 샘플의 수, N^{01}는 분류기 c_i는 틀리고 c_k는 맞추는 샘플의 수, N^{10}는 c_i는 맞추고 c_k는 틀리는 샘플의 수를 뜻한다. CD 값이 크면 분류기 간의 거리가 크게 된다.

$$CD_{ij} = 1 - prob(c_i\ fails, c_j\ fails) = 1 - \frac{N^{00}}{N^{11} + N^{00} + N^{01} + N^{10}}$$

3.2 군집화 알고리즘

이 논문에서 사용한 군집화 알고리즘은 k-means 방법과 비슷하다. 그림 2는 군집화 알고리즘의 전체 구조이다. D는 초기 분류기 풀의 크기이다.

C_i는 i 번째 군집, $1 \le i \le k$이고, c_j는 j번째 분류기, 그리고 z_i는 C_i 의 대표를 나타낸다.

InitCenter(): 분류기들 중에서 임의로 k 개를 선택하여 z_i로 할당한다.

AssignToClusters(): 분류기 각각을 군집 대표와 거

```
입력: D개의 인식 결과 벡터, 원하는 군집 개수 k
출력: k개의 군집을 대표하는 분류기
{
 InitCenter();
 AssignToClusters();

 while(not stopping-condition) {
  ChangeCenter();
  AssignToClusters();
 }
}
```

그림 2 군집화 알고리즘의 전체 구조

리 CD를 계산하여 가장 가까운 군집에 할당한다.

ChangeCenter(): 군집 C_i에 대해 대표 분류기를 선정한다. C_i 속하는 분류기 c_j에 대하여 C_i에 속하는 나머지 모든 분류기 c_k와의 CD_{jk}를 계산하고 이의 평균을 내어 가장 작은 평균값을 갖는 분류기 c_j를 z_i로 한다.

위에서 설명한 군집화 알고리즘이 k-means와 다른 점은 군집 C_i의 대표 샘플을 선정하는 방법에 있다. k-means는 C_i에 속하는 특징 벡터의 평균을 구하여 그것을 대표 벡터로 삼는다. 하지만 분류기를 군집화하는 경우에는 분류기가 인식 벡터로 표현되므로 평균이라는 개념을 도입할 수 없다. 따라서 C_i에 속하는 분류기들 중에서 ChangeCenter()가 사용하는 방법으로 대표를 선정한다.

4. 유전 알고리즘에 의한 분류기 앙상블 선택

그림 3은 분류기 앙상블 선택을 위한 유전 알고리즘의 구조이다. k는 다음 세대로 진화할 때 대치되는 자식 해(offspring)의 수이다. k가 해집단의 크기와 같을 때는 세대형(generational) 유전 알고리즘이 되고, k가 1일 때는 안정상태(steady-state) 유전 알고리즘이 된다.

```
초기 해집단 생성;

repeat {
 for i=1 to k {
  두 염색체 p1, p2 선택;
  offspring_i = crossover(p1, p2);
  offspring_i = mutation(offspring_i);
  if (HYBRID) local_improve(offspring_i);
 }
 offspring_1, ..., offspring_k를 대치;
} until (정지 조건 만족);
현재까지 최상의 염색체를 return;
```

그림 3 유전 알고리즘의 전체 구조(HYBRID 불린 변수가 FALSE이면 SGA이고 TRUE이면 HGA임)

다음 세대로의 진화 방법은 k=1인 안정형을 사용하였다. 유전 알고리즘은 매개 변수 설정에 따라 해 품질이 크게 영향을 받으므로, 가능한 자세히 알고리즘과 매개변수에 대해 기술한다. 먼저 단순 유전 알고리즘을 설명하고, 지역 탐색 연산을 추가한 혼합 유전 알고리즘을 설명한다. 단순 유전 알고리즘에서는 그림 3의 불린 변수 HYBRID를 FALSE로 설정하여 local_search() 연산을 수행하지 않고, 혼합 유전 알고리즘은 HYBRID를 TRUE로 설정하면 된다.

4.1 단순 유전 알고리즘(SGA: Simple GA)

(1) 염색체 표현

염색체의 길이는 분류기 풀의 크기인 D이다. 염색체는 이진열(binary string)로 표현되며 '1'은 분류기가 선택되었음을 의미하고 '0'은 배제되었음을 의미한다. 따라서 '1'인 유전자의 개수는 선택된 분류기의 개수인 d와 같아야 한다. 예를 들어 하나의 해가 1000100101(D=10인 경우) 일 때, 1번, 5번, 8번, 10번 분류기가 선택되어 있고 분류기 앙상블 크기는 4이다.

(2) 초기해 집단

각 염색체는 임의수를 생성하여 d 개의 유전자(gene)가 '1'의 값을 갖도록 초기화 한다. 각 유전자에 대해 [0, 1] 사이의 임의의 r을 생성하고, $r < d/D$이면 1, 그렇지 않으면 0 값을 부여한다. 이렇게 하면 평균적으로 d 개의 유전자가 1을 갖게 되는데, 모든 해가 정확히 d 개를 갖지는 않는다. 따라서 이런 경우에는 임의의 유전자를 선택하여 0-1 또는 1-0으로 변환시켜 1인 유전자가 d 개가 되도록 한다. 해집단(population)의 크기는 20으로 한다.

(3) 적합도 계산

분류기 앙상블에 속한 분류기들에 대해 투표(voting) 방법에 의해 결합 인식률을 구하고 이를 이 해의 품질로 한다. 이 품질을 아래에서 설명하는 순위기반 방법을 이용하여 적합도(fitness)로 변환한다. 즉 해집단 내의 해들을 품질(결합 인식률)에 따라 정렬한 후, i 번째 해의 적합도를 아래 함수에 따라 배정한다. f_i는 i 번째 해의 적합도, n은 해집단의 크기이고, min과 max는 사용자가 설정하는 매개 변수이다.

$$f_i = \max + (i-1) \times (\min - \max)/(n-1)$$

결국 가장 좋은 해는 max 값, 가장 나쁜 해는 min 값을 갖게 되고, 모든 해는 그 사이 값을 갖게 된다. min과 max 값의 차이가 클수록 선택압(selection pressure)이 높아진다. min=0.1, max=1.0을 사용하였다.

(4) 선택

룰렛-휠 방법에 의해 부모 염색체를 선택한다. 이 방법에서는 우선 각 해에 대해 $f'_i = f_i / F$을 계산한다. F

는 모든 해의 적합도를 합한 값이다. 따라서 $\sum_{i=1}^{n} f'_i = 1$ 이 된다. n은 해집단의 크기이다. [0, 1] 공간을 n 개의 해에 대해 f_i에 따라 영역을 배정한다. i 번째 해는 $\left[\sum_{k=1}^{i-1} f'_k, \sum_{k=1}^{i} f'_k\right]$영역을 배정 받는다. 단 i=1인 첫 번째 해는 [0, f'_1]을 배정받는다. 부모 염색체 선택을 위해서는 [0, 1]사이의 임의수를 생성하고 이 임의수를 포함하는 영역을 가진 해를 선택하면 된다.

(5) 교배와 돌연변이

m 개의 자름선을 임의로 선택한 후, 두 부모 염색체를 교차 시켜 자식 염색체를 생성한다. m은 3으로 하였다. 그 다음 두 개의 자식 염색체는 돌연변이 연산을 거친다. 이때 염색체의 모든 유전자에 대해 [0, 1] 사이에 있는 임의의 수 r을 생성하여 r이 돌연변이 확률 (p_m= 0.1)보다 작은 경우에는 해당 유전자의 값을 0은 1로 1은 0으로 변환한다.

교차 연산과 돌연 변이 연산은 임의수에 의존하므로 연산이 끝난 후 '1'인 유전자의 수가 d와 항상 일치하지는 않는다. 따라서 돌연 변이 후, '1'인 유전자의 수가 d와 다를 경우 유전자를 임의로 선택하여 1-0 또는 0-1 변환을 시켜 '1'인 유전자의 수가 d가 되도록 수선(repair) 연산을 수행한다.

(6) 대치

두개의 자식 중에 우수한 자식을 선택하여 두 부모보다 우월하면 자신과 비슷한 부모를 대치하고, 두 부모 사이라면 열세한 부모를 대치한다. 자식이 두 부모보다 열세하다면 해 집단에서 가장 열세한 염색체와 대치한다.

이 방법은 preselection이라 불리는데 다양성 확보와 관련이 있다. 이는 해집단에서 자신과 닮았을 가능성이 가장 높은 해가 부모해들이므로 다른 해들보다 부모해 중의 하나를 제거함으로써 해집단의 다양성을 오래 유지시키는데 도움이 된다.

(7) 종료

유전 과정의 세대 수가 미리 정한 최대 세대 수 T에 도달하면 중단한다. 유전알고리즘에 의해 찾은 최대값이 일정 세대 수가 지나도 더 이상 변하지 않을 때를 수렴되었다고 가정하고, 최대값이 변하지 않기 시작하는 세대수를 최대 세대 수를 실험적으로 추정하여 T로 정하였다.

4.2 혼합 유전 알고리즘 (HGA: Hybrid GA)

유전 알고리즘은 교배나 변이 연산을 통해 지역 최적해에 빠지는 것을 방지할 수 있고, 선택 압력을 적절하게 조절하여 문제 공간의 넓은 범위를 탐색할 수 있다. 그러나 유전 알고리즘은 지역 최적해 근처에서 미세 조정력(fine tuning)이 약하기 때문에 긴 실행 시간을 필요로 한다. 이러한 이유로 TSP 문제, 그래프 분할 문제, 영상 압축 문제, 특징 선택 문제 등 많은 응용에서 단순 유전 알고리즘에 미세 조정력을 향상시키기 위해 혼합 유전 알고리즘을 적용하였다[문병로03].

교배와 변이 연산을 통해 자식 염색체는 부모 염색체로부터 좋은 특성들을 상속받는다. 이렇게 생성된 자식 염색체는 그들의 부모보다 우수하거나 열세할 수 있다. 혼합 유전 알고리즘은 자식 염색체들을 대치 연산 전에 지역 탐색 연산을 통해서 품질을 향상 시킨다.

지역 탐색 연산의 중요 관점은 지역 향상을 위해 적절한 연산을 선택하는 것이다. 이 논문에서 사용한 지역 탐색 연산의 구조는 아래와 같다. 단순 유전 알고리즘에서는 교배와 돌연변이 연산 후에 1인 유전자 수를 d개로 맞추기 위해 수선 연산을 수행하였는데, 여기서는 지역 탐색 연산 local_improve()가 이를 담당하므로 수선 연산을 따로 수행하지 않는다.

```
local_improve(offspring) {
  g = offspring의 '1'인 유전자의 수;
  switch {
     case g=d : RippleRem(r); RippleAdd(r);
     case g<d : for(i=0; i<d-g; i++) RippleAdd(r);
     case g>d : for(i=0; i<g-d; i++) RippleRem(r);
  }
}
```

RippleRem(r)과 RippleAdd(r)은 특징 선택 문제를 위해 Oh 등에 의해서 처음 제안되었다[Oh04]. 분류기 앙상블 선택 문제는 특징 선택 문제와 같은 문제로 볼 수 있다. 단지 선택 대상이 특징이 아니라 분류기라는 점과 평가 함수가 결합 알고리즘이라는 점이 다른 뿐이다. 따라서 이 논문에서는 특징 선택 문제에서 이미 우수한 성능을 보인 RippleRem(r)과 RippleAdd(r) 연산을 사용하였다.

rem(): '1'인 유전자 중에 가장 의미 없는 (그것 하나를 뺐을 때 성능 저하가 가장 적은) 유전자를 찾아 '0'으로 한다.

add(): '0'인 유전자 중에 가장 의미 있는 (그것 하나를 더했을 때 성능 향상이 가장 큰) 유전자를 찾아 '1'로 한다.

REM(k): rem() 연산을 k번 반복한다.

ADD(k): add() 연산을 k번 반복한다.

RippleRem(r) ≡ { REM(r); ADD(r-1); }, r >= 1

RippleAdd(r) ≡ { ADD(r); REM(r-1); }, r >= 1

rem 연산은 현재 선택된 앙상블에서 가장 의미 없는

분류기를 하나 제거하는 역할을 한다. 반대로 add 연산은 선택되지 않은 분류기들의 집합에서 가장 의미 있는 분류기를 선택하여 앙상블에 추가해주는 역할을 한다. g는 현재 해가 갖는 선택된 분류기의 수를 의미한다. 원하는 분류기 앙상블의 크기가 d이므로 $|d-g|$ 만큼의 반복을 통해 분류기를 추가 또는 제거해준다.

r은 탐색의 깊이를 조절하는 인자로 지역 탐색 성능 향상의 강도를 조절한다. r은 rem과 add 연산의 실행 횟수에 직접적인 영향을 준다. r이 클수록 넓은 영역을 탐색하고 그에 따른 시간 비용도 증가한다. 그림 4는 r=2, g=3이고 d=5일 때의 지역 탐색 연산의 동작 예이다. for 반복문에 의해 RippleAdd(2)을 2번 실행하게 된다.

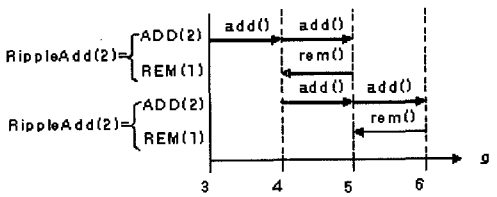

그림 4 지역 탐색 연산 동작 예

5. 실험과 분석

5.1 실험 환경

실험의 객관성을 위하여, 세계적으로 표준 데이타베이스로 인정된 필기 숫자 데이타베이스 두 종류, CENPAMI와 MNIST 데이타베이스를 사용한다. 표 1은 실험에 사용한 필기 숫자 데이타 집합을 보여준다. CENPARMI 숫자 데이타 집합은 CENPARMI, Concordia University에서 만들었으며, 총 4000개의 훈련 샘플과 2000개의 테스트 샘플로 나뉘어져 있다[Liu03]. 앙상블 실험을 위해 훈련 샘플 중 3000개를 분류기의 훈련을 위해 사용 하였고, 1000개는 분류기의 앙상블 선택을 위해 검증(validation) 샘플로 사용 하였으며, 테스트 샘플은 분류기 앙상블의 성능 측정을 위해 사용하였다.

MNIST 숫자 데이타 집합은 NIST 데이타 집합 SD3와 SD7로부터 추출되어 만들어졌다[Liu03]. 훈련 데이타 샘플 60000개와 테스트 샘플 10000개로 구성되어 있으며, 훈련 데이타 샘플로부터 분류기 훈련을 위해 40000개, 검증을 위해 20000개로 나누어 사용하였다.

5.2 초기 분류기 풀의 성능

표 1 실험에 사용한 데이타 집합

	훈련 샘플	검증 샘플	테스트 샘플
CENPARMI	3000	1000	2000
MNIST	40000	20000	10000

분류기의 다양성을 위해서 각 데이타 집합에서 3가지 특징을 추출하였다. 또한 특징 부분집합을 생성하여 각 분류기의 다양성을 높이고 많은 수의 분류기를 생성하였다. MLP, KNN, SVM의 각 분류기 종류마다 3가지 특징 DDD, AGD, MESH를 사용하였고[Oh98], 특징별로 특징 부분집합 50개씩을 임의 선택하여 총 3*3*50=450개의 분류기를 만들었다. 특징 부분집합의 크기는 원래 특징 집합의 70~90%로 하였다.

각 분류기마다 파라미터는 고정시켰다. MLP의 생성을 위해 부류마다 고유의 인식기를 갖는 OCOC (One-class-One-Classifier) 분류 알고리즘을 사용하였다[이진선05]. 입력 층, 은닉 층, 출력 층의 3 계층 모델을 사용하였고, 훈련 방법으로는 backpropagation 알고리즘을 사용하였다. 각 OCOC는 입력 노드는 특징 부분집합의 크기로 설정하였고, 은닉 노드는 4 개, 출력 노드는 2개로 설계하였다. KNN은 1-NN 분류 알고리즘을 사용하였다. SVM은 SVM-light를 사용하였다[SVMLIGHT07]. 커널은 polynomial$(s\,a\times b+c)^d$를 사용하였고, d는 2로 c는 6으로 사용하였다.

표 2와 3은 생성된 초기 분류기 풀의 성능을 보여준다. 표 2를 살펴보면, CENPARMI 데이타 집합의 초기 분류기 풀에는 테스트 집합에 대해 97.9% 인식률을 갖는 분류기(DDD를 사용한 SVM)부터 91.55% 인식률을 갖는 분류기(AGD를 사용한 KNN)까지 분류기간의 우열이 큼을 알 수 있다.

표 3을 살펴보면, MNIST 데이타 집합의 초기 분류기 풀에는 테스트 집합에 대해 98.85% 인식률을 갖는 분류기(AGD를 사용한 SVM)부터 94.11% 인식률을 갖는 분류기(DDD를 사용한 KNN)까지 가지고 있어, 분류기간의 우열이 역시 큼을 알 수 있다.

5.3 기존 알고리즘과 성능 비교

아래에서 비교 대상인 알고리즘을 설명한다. 모든 알고리즘은 5.2절에서 만든 분류기 풀을 입력으로 받는다. 분류기 앙상블의 결합은 다수 투표(majority voting) 방법을 사용하였다.

(1) 순위에 따라 선택

- Ranking_Simple: 분류기 풀의 분류기들을 주어진 기준에 따라 정렬한 후, 가장 좋은 d 개의 분류기를 선택한다. 여기서 d는 앙상블 크기이다. 기준으로는 개별 분류기의 인식률을 사용한다.
- Ranking_ByClass: 분류 알고리즘 별로 (SVM, KNN, MLP) 분류기들을 정렬한 후 가장 좋은 것들을 d/3개씩 선택한다. 앞의 최상 선택보다 다양성 면에서 유리하다.

(2) 군집화 방법

- Clustering_CD: 3장에 있는 그림 2의 알고리즘을 사

표 2 CENPARMI 데이타 집합의 초기 분류기 풀의 성능

분류기종류		MLP	KNN	SVM
특징	샘플	평균(최소,최대)	평균(최소,최대)	평균(최소,최대)
DDD	검증	96.70(96.20,97.20)	94.17(93.70,94.90)	98.38(98.10,98.60)
	테스트	96.59(96.50,96.80)	93.59(93.40,93.80)	97.61(97.15,97.90)
AGD	검증	96.28(95.90,96.70)	91.75(91.30,92.00)	97.54(97.10,98.00)
	테스트	95.58(95.50,95.85)	91.80(91.55,92.00)	96.18(96.10,96.50)
MESH	검증	93.64(92.60,94.60)	95.02(94.50,95.60)	96.26(95.30,97.10)
	테스트	92.23(91.90,93.25)	94.40(94.35,94.50)	95.13(94.35,95.70)

표 3 MNIST 데이타 집합의 초기 분류기 풀의 성능

분류기종류		MLP	KNN	SVM
특징	샘플	평균(최소,최대)	평균(최소,최대)	평균(최소,최대)
DDD	검증	96.97(96.67,97.15)	94.21(93.96,94.42)	97.77(96.50,97.90)
	테스트	97.26(97.26,97.35)	94.26(94.11,94.45)	97.75(96.97,97.85)
AGD	검증	98.06(97.95,98.15)	95.30(94.84,95.79)	98.64(98.56,98.75)
	테스트	98.56(98.53,98.69)	95.61(95.30,96.28)	98.70(98.55,98.85)
MESH	검증	94.98(94.63,95.26)	97.17(97.07,97.33)	98.10(98.04,98.16)
	테스트	95.92(95.78,96.20)	97.03(96.94,97.21)	98.15(98.10,98.25)

용한다. 원하는 군집의 개수 k를 d로 설정한다.

(3) 유전 알고리즘

- SGA_AfterRanking: 우선 인식률 순위에 따라 90개의 분류기를 선택한 후, 90개에서 d개를 4.1절의 단순 유전 알고리즘으로 찾는다.

(4) 제안한 알고리즘

- Multistage_SGA: 3장의 군집화 알고리즘으로 90개의 분류기를 찾고, 90개에서 d개를 4.1절의 단순 유전 알고리즘으로 찾는다.
- Multistage_HGA: 3장의 군집화 알고리즘으로 90개의 분류기를 찾고, 90개에서 d개를 4.2절의 혼합 유전 알고리즘으로 찾는다.

이 절에서는 우선 앞의 다섯 개 알고리즘을 비교 분석한다. 5.4 절에서는 Multistage_SGA와 Multistage_HGA에 초점을 맞추어 비교 분석함으로써 HGA가 탐색 성능을 얼마나 향상시키는지를 알아본다.

표 4는 CENPARMI 데이타 집합에 대하여 네 개의 기존 알고리즘과 (Ranking_Simple, Ranking_ByClass, Clustering_CD, SGA_AfterRanking) 제안한 Multistage_SGA의 성능을 보여준다. 같은 앙상블 크기(d)에 대하여 가장 좋은 해를 찾은 알고리즘에 대해서는 굵은 글씨체로 표시하였다. 그림 5는 표 4를 알아보기 쉽도록 그래프로 표현한 것이다.

같은 앙상블 크기에서 가장 좋은 해를 찾은 경우를 세어보았는데 Multistage_SGA 방법이 14번, 그 다음으로 Clustering_CD 방법이 10번으로 많았다. 또한 Multistage_SGA 방법이 가장 좋은 해 (d=6일 때 98.35%)를 찾았음을 볼 수 있다.

인식률의 평균을 계산해보면, Multistage_SGA 방법의 인식률은 Ranking_Simple 방법에 비교해 평균 0.34% 우수했고, Ranking_ByClass 방법에 비교하여 평균 0.27%, Clustering_CD 방법에 비해 평균 0.03% 우수하였다. 그리고 SGA_AfterRanking 방법의 인식률에 비해 평균 0.19% 우수하였다.

제안한 알고리즘을 제외하고, 기존의 네 개 알고리즘에 대해 성능을 비교해 볼 수 있다. 인식률의 평균을 계산해보면, Clustering_CD 방법이 98.05%로 가장 높았고, SGA_AfterRanking 방법이 97.89%로 두 번째로 높았으며, 세 번째 순위는 97.80%로 Ranking_ByClass 방법이었다. 마지막 순위는 평균 인식률 97.73%로 Ranking_Simple 방법이었다. 또한 같은 앙상블 크기에 대하여 네 개 알고리즘을 비교하여 가장 좋은 해를 찾은 경우를 세어보았는데, Clustering_CD 방법이 18번으로 가장 많았고, SGA_AfterRanking 방법이 6번, Ranking_ByClass 방법이 3번이었고 Ranking_Simple 방법은 1번이었다.

표 5는 MNIST 데이타 집합에 대한 성능 비교를 보여준다. 그림 6은 표 5를 그래프로 표현한 것이다. 가장 좋은 해를 찾은 경우를 세어보았는데 Multistage_SGA 방법이 20번, 그 다음으로 Clustering_CD 방법이 5번으로 많았다. 또한 Multistage_SGA 방법이 앙상블의 가장 좋은 해(d=30일 때와 d=42일 때 99.02%)를 찾았다.

인식률의 평균을 계산해보면, Multistage_SGA 방법의 인식률은 Ranking_Simple 방법에 비교해 평균 0.17% 우수했고, Ranking_ByClass 방법에 비교하여 평균 0.07%, Clustering_CD 방법에 비해 평균 0.12% 우수하였다.

표 4 CENPARMI 데이타 집합에 대한 알고리즘들의 인식률

(*d*: 앙상블 크기, 표기: (검증 데이타 인식률)테스트 데이타 인식률)

d	Ranking_Simple	Ranking_ByClass	Clustering_CD	SGA_AfterRanking	Multistage_SGA
2	(98.60)**97.65**	(98.60)**97.65**	(98.50)97.55	(98.60)**97.65**	(98.60)**97.65**
4	(98.60)97.65	(98.50)**97.85**	(98.60)97.75	(98.70)**97.85**	(99.00)97.55
6	(98.50)97.75	(98.60)98.00	(98.70)98.10	(98.60)97.75	(99.20)**98.35**
8	(98.50)97.80	(98.50)98.00	(98.90)**98.25**	(98.70)97.75	(99.10)98.00
10	(98.50)97.75	(98.50)98.00	(98.90)98.00	(98.60)98.00	(99.20)**98.20**
12	(98.60)97.70	(98.40)97.70	(99.00)**98.20**	(98.70)97.80	(99.20)98.00
14	(98.50)97.70	(98.40)97.65	(99.00)**98.10**	(98.70)97.90	(99.20)98.05
16	(98.50)97.70	(98.40)97.65	(99.10)98.00	(98.60)97.75	(99.20)**98.05**
18	(98.50)97.70	(98.40)97.75	(98.80)98.10	(98.60)97.85	(99.20)**98.15**
20	(98.50)97.70	(98.40)97.75	(99.00)**98.20**	(98.60)97.90	(99.20)98.15
22	(98.50)97.80	(98.40)97.80	(98.90)97.95	(98.60)98.00	(99.20)**98.25**
24	(98.50)97.75	(98.40)97.85	(99.00)**98.15**	(98.70)97.70	(99.20)98.05
26	(98.50)97.75	(98.40)97.85	(98.90)**98.25**	(98.60)97.95	(99.10)98.05
28	(98.50)97.75	(98.40)97.80	(99.00)98.00	(98.60)97.95	(99.20)**98.20**
30	(98.50)97.75	(98.30)97.75	(98.90)97.95	(98.60)97.90	(99.20)**98.30**
32	(98.50)97.70	(98.30)97.80	(98.80)**98.10**	(98.60)97.95	(99.10)**98.10**
34	(98.50)97.75	(98.40)97.80	(98.90)**98.15**	(98.70)97.80	(99.10)98.10
36	(98.50)97.75	(98.30)97.85	(98.90)**98.05**	(98.60)97.95	(99.10)98.00
38	(98.50)97.75	(98.30)97.80	(98.90)98.00	(98.60)97.95	(99.20)**98.10**
40	(98.50)97.75	(98.40)97.80	(98.90)**98.10**	(98.60)98.00	(99.10)98.00
42	(98.50)97.75	(98.30)97.75	(98.90)98.15	(98.60)97.95	(99.10)**98.20**
44	(98.50)97.75	(98.30)97.80	(98.90)98.10	(98.60)98.00	(99.10)**98.15**
46	(98.50)97.75	(98.30)97.80	(99.00)98.00	(98.60)98.00	(99.10)**98.15**
48	(98.40)97.70	(98.30)97.80	(99.00)97.90	(98.60)97.95	(99.10)**98.15**

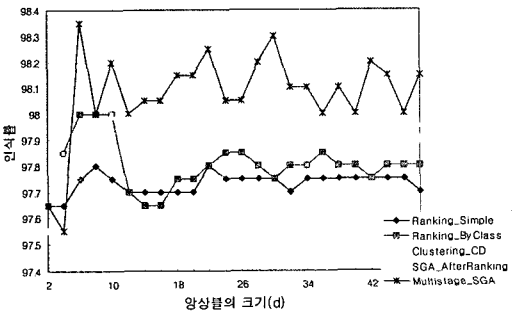

그림 5 CENPARMI 데이터 집합에 대한 앙상블 알고리즘들의 성능 비교

그리고 SGA_AfterRanking 방법의 인식률에 비해 평균 0.07% 우수하였다.

제안한 알고리즘을 제외하고, 기존 네 개 알고리즘에 대한 성능 비교에 대해 언급하겠다. Ranking_Simple 방법은 나머지 세 알고리즘과 비해 평균 인식률도 제일 낮았고, 좋은 해를 찾은 경우는 한 번도 없었다. 나머지 Ranking_ByClass 방법, Clustering_CD 방법, SGA_AfterRanking 방법을 비교하여 같은 앙상블의 크기에 대하여 가장 좋은 해를 찾은 경우를 세어보았는데, Ranking_ByClass 방법이 9번 Clustering_CD 방법이 8번, SGA_AfterRanking 방법이 8번으로 세 알고리즘이 비슷한 경우의 수로 좋은 해를 찾았다.

Clustering_CD 방법은 나머지 세 알고리즘에 비해 상대적으로 우수한 해를 찾은 경우(*d*=32일 때와 *d*=38일 때 98.96%, *d*=14일 때 98.94%)가 있었으나 성능이 제일 낮은 해를 찾은 경우(*d*=34일 때 98.57%, *d*=18일 때 98.63%)도 있었고, 찾은 해의 품질에 좋고 나쁨에 변동이 많았다. 이런 이유로 평균 인식률을 계산해 보면, SGA_AfterRanking 방법과 Ranking_ByClass 방법이 98.87%로 동점이었고, Clustering_CD 방법은 98.82%로 더 낮은 성능을 보였다.

앞의 CENPARMI 데이타 집합과 MNIST 데이타 집합의 기존 알고리즘과의 비교 실험 결과를 통해 우리가 제안한 다단계 선택 방법이 기존 방법에 비해 우수함을 알 수 있다.

5.4 Multistage_SGA와 Multistage_HGA 성능 비교

이 절에서는 이 논문에서 제안한 Multistage_SGA와 Multistage_HGA의 탐색 능력을 비교한다. 표 6은 CENPARMI 데이타 집합과 MNIST 데이타 집합에 대하여 Multistage_SGA와 Multistage_HGA 방법으로 선택한 분류기 앙상블의 성능이다.

표 5 MNIST 데이타 집합에 대한 알고리즘들의 인식률
(*d*: 앙상블 크기, 표기: (검증 데이타 인식률)테스트 데이타 인식률)

d	Ranking_Simple	Ranking_ByClass	Clustering_CD	SGA_AfterRanking	Multistage_SGA
2	(98.75)98.67	(98.75)98.67	(98.69)**98.72**	(98.75)98.67	(98.75)98.67
4	(98.76)98.72	(98.83)98.77	(98.81)98.74	(98.87)98.78	(99.03)**98.92**
6	(98.76)98.75	(98.88)98.92	(98.76)98.78	(98.92)98.87	(99.06)**98.93**
8	(98.79)98.77	(98.83)98.88	(98.80)98.68	(98.93)98.85	(99.07)**98.91**
10	(98.77)98.80	(98.86)98.87	(98.99)98.80	(98.96)98.89	(99.14)**98.98**
12	(98.77)98.80	(98.89)98.87	(99.09)98.91	(98.94)98.87	(99.09)**98.93**
14	(98.76)98.80	(98.86)98.87	(99.00)**98.94**	(98.94)98.83	(99.10)**98.94**
16	(98.74)98.78	(98.90)98.87	(98.98)**98.90**	(98.96)98.87	(99.07)98.88
18	(98.74)98.79	(98.91)98.87	(98.80)98.63	(98.98)98.90	(99.12)**98.94**
20	(98.73)98.77	(98.89)98.89	(99.01)98.92	(98.97)98.90	(99.12)**98.98**
22	(98.74)98.78	(98.91)98.90	(98.96)98.86	(98.97)98.84	(99.10)**98.96**
24	(98.76)98.76	(98.92)98.89	(98.92)98.80	(98.97)98.90	(99.10)**98.92**
26	(98.76)98.75	(98.92)98.91	(99.05)98.90	(98.96)98.87	(99.09)**99.00**
28	(98.75)98.77	(98.92)98.90	(98.92)98.87	(98.97)98.89	(99.13)**98.92**
30	(98.74)98.75	(98.92)98.88	(98.87)98.81	(98.96)98.88	(99.08)**99.02**
32	(98.74)98.75	(98.92)98.90	(98.99)**98.96**	(98.97)98.90	(99.07)**98.96**
34	(98.72)98.77	(98.91)98.92	(98.67)98.57	(98.96)98.90	(99.11)**98.96**
36	(98.72)98.77	(98.92)**98.90**	(98.90)98.88	(98.99)98.85	(99.08)98.85
38	(98.73)98.77	(98.91)98.88	(99.05)**98.96**	(98.97)98.89	(99.09)98.94
40	(98.73)98.79	(98.92)98.89	(98.98)98.95	(98.98)98.87	(99.07)**99.01**
42	(98.73)98.81	(98.92)98.89	(98.91)98.79	(98.96)98.92	(99.08)**99.02**
44	(98.74)98.78	(98.89)98.88	(98.83)98.81	(98.95)98.94	(99.07)**98.96**
46	(98.74)98.77	(98.91)98.88	(98.89)98.75	(98.96)98.90	(99.06)**99.00**
48	(98.73)98.77	(98.92)98.89	(98.95)98.76	(98.95)98.87	(99.04)**98.96**

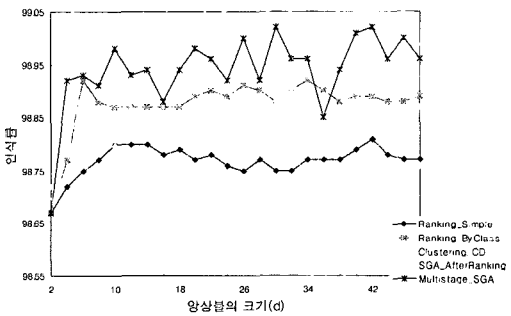

그림 6 MNIST 데이타 집합에 대한 알고리즘들의 성능 비교

CENPARMI 데이타 집합과 MNIST 데이타 집합에 대하여 앙상블의 크기에 관계없이 모든 경우에서 Multistage_HGA가 Multistage_SGA보다 좋은 해를 찾았다. 그림 7과 그림 8은 표 6의 내용을 그래프로 그린 것이다.

그림 7은 CENPARMI 데이타 집합에 대하여 Multistage_SGA와 Multistage_HGA 방법으로 선택한 분류기 앙상블의 성능이다. 같은 앙상블의 크기에 대하여 반복 실험 횟수는 20회로 하였다. 최고값은 20회 실험 중 가장 높은 값을 표시한 것이다.

먼저 Multistage_HGA와 Multistage_SGA의 평균값 분포를 비교해 볼 때, 모든 앙상블의 크기에서 Multistage_HGA가 Multistage_SGA에 비해 우수하다. Multistage_HGA와 Multistage_SGA의 평균값의 차이를 살펴보면 앙상블의 크기가 14일 때 가장 큰 차를 보이고 값의 차는 0.27%이다. Multistage_HGA 방법으로 찾은 최고값은 앙상블 크기 10일 때 98.6%이었고, Multistage_SGA 방법으로 찾은 최고값은 앙상블의 크기 8일 때 98.35%였다.

앙상블의 크기에 따른 인식률의 경향을 살펴보면, 앙상블의 크기가 커짐에 따라 인식률이 증가하다가 앙상블의 크기 25 이후에는 점점 인식률이 감소하는 경향을 보인다. Multistage_HGA의 평균값이 98.3%를 넘는 앙상블 크기를 살펴보니 앙상블의 크기가 8~23인 곳에서 나타났으며, 이를 통해 CENPARMI 집합에 대한 최적의 앙상블 크기를 찾을 수 있었다.

그림 8은 MNIST 데이타 집합에 대하여 Multistage_SGA와 Multistage_HGA 방법으로 선택한 분류기 앙상블의 성능이다. 먼저 Multistage_HGA와 Multistage_SGA의 평균값 분포를 비교해 볼 때, 모든 앙상블의 크기에서 Multistage_HGA가 Multistage_SGA에

표 6 테스트 집합에 대한 Multistage_SGA와 Multistage_HGA의 성능 비교
(*d* 각각에 대해 20회 반복 수행) (표기: 평균 인식률 (최고 인식률))

	CENPARMI				MNIST			
d	Multistage_SGA		Multistage_HGA		Multistage_SGA		Multistage_HGA	
2	97.65	(97.65)	97.69	(97.80)	98.67	(98.67)	98.69	(98.76)
4	97.87	(98.20)	98.19	(98.30)	98.93	(98.99)	98.95	(99.01)
6	98.02	(98.25)	98.24	(98.40)	98.94	(99.00)	98.98	(99.02)
8	98.09	(98.35)	98.32	(98.50)	98.94	(99.00)	99.00	(99.04)
10	98.03	(98.25)	98.35	(98.60)	98.97	(99.04)	99.01	(99.06)
12	98.07	(98.35)	98.38	(98.55)	98.97	(99.03)	99.02	(99.08)
14	98.07	(98.30)	98.34	(98.45)	98.96	(99.01)	99.03	(99.06)
16	98.10	(98.30)	98.31	(98.50)	98.96	(99.02)	99.03	(99.08)
18	98.10	(98.25)	98.32	(98.50)	98.99	(99.04)	99.05	(99.11)
20	98.08	(98.35)	98.32	(98.45)	98.99	(99.05)	99.06	(99.10)
22	98.12	(98.35)	98.31	(98.50)	98.99	(99.05)	99.05	(99.08)
24	98.07	(98.25)	98.27	(98.40)	99.01	(99.04)	99.05	(99.07)
26	98.13	(98.30)	98.28	(98.40)	98.98	(99.02)	99.05	(99.08)
28	98.12	(98.25)	98.26	(98.40)	98.98	(99.04)	99.05	(99.09)
30	98.10	(98.20)	98.24	(98.40)	98.97	(99.02)	99.05	(99.08)
32	98.09	(98.30)	98.22	(98.30)	98.99	(99.04)	99.05	(99.10)
34	98.07	(98.20)	98.23	(98.30)	98.98	(99.03)	99.05	(99.09)
36	98.12	(98.25)	98.25	(98.30)	98.99	(99.03)	99.04	(99.11)
38	98.10	(98.20)	98.23	(98.30)	98.99	(99.01)	99.04	(99.09)
40	98.07	(98.20)	98.23	(98.35)	98.97	(99.03)	99.04	(99.07)
42	98.09	(98.20)	98.22	(98.35)	98.96	(99.00)	99.03	(99.06)
44	98.09	(98.20)	98.21	(98.30)	98.97	(99.02)	99.02	(99.06)
46	98.06	(98.20)	98.20	(98.25)	98.97	(99.02)	99.02	(99.06)
48	98.07	(98.20)	98.19	(98.30)	98.96	(99.03)	99.02	(99.06)

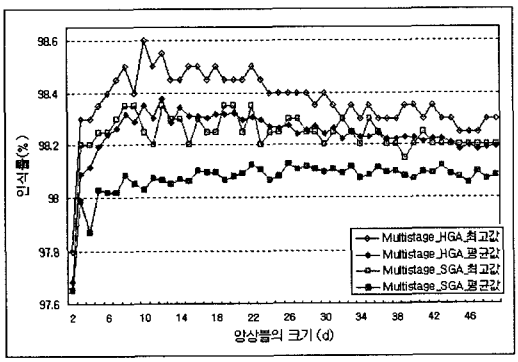

그림 7 CENPARMI 데이타에 대한 Multistage_SGA와 Multistage_HGA의 성능 비교

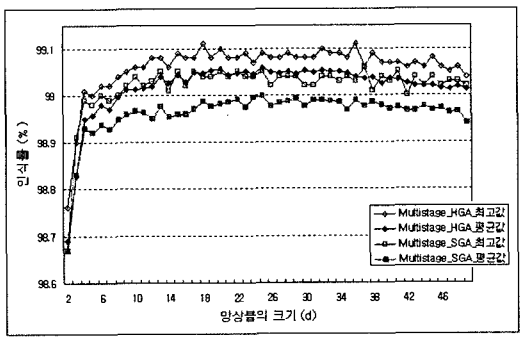

그림 8 MNIST 데이타에 대한 Multistage_SGA와 Multistage_HGA의 성능 비교

비해 우수하다. Multistage_HGA와 Multistage_SGA의 평균값의 차이를 살펴보면 앙상블의 크기가 15일 때 가장 큰 차를 보이고 값의 차는 0.13%이다. Multistage_HGA 방법으로 찾은 최고값은 앙상블 크기 18일 때 99.11%이었고, Multistage_SGA 방법으로 찾은 최고값은 앙상블의 크기 20일 때 99.05%였다.

앙상블 크기에 따른 인식률의 변화는 앙상블 크기가 증가함에 따라 인식률도 증가하다가 앙상블의 크기 35 이후로는 앙상블의 인식률이 점점 감소하는 경향을 보인다. Multistage_HGA의 평균값이 99.05%를 넘는 앙상블 크기를 살펴보니 앙상블의 크기가 17~35인 곳에서 나타났으며, 이를 통해 MNIST 집합에 대한 최적의 앙상블 크기를 찾을 수 있었다.

표 7은 기존 알고리즘과 이 논문에서 제안한 알고리즘을 구분하여 가장 좋은 해를 찾은 경우를 보여준다.

표 7 가장 좋은 해 비교(테스트 데이타에 대한 정인식률)

	기존 알고리즘	제안한 알고리즘
CENPARMI	98.25% (d=8) (Clustering_CD)	98.60% (d=10) (Multistage_HGA)
MNIST	98.96% (d=32) (Clustering_CD)	99.11% (d=18) (Multistage_HGA)

표 7의 '가장 좋은 해'는 앙상블의 크기에 상관없이 인식률이 가장 좋은 해를 의미한다.

기존의 앙상블 선택 알고리즘 중 가장 좋은 해는 Clustering_CD 방법이 찾은 것으로, CENPARMI 데이타의 경우에는 인식률 98.25%이였고, MNIST 데이타의 경우에는 인식률 98.96%이었다.

본 논문에서 제안한 방법을 적용하여 찾은 가장 좋은 해는 CENPARMI 데이타에 경우에는 인식률 98.60%이었고, MNIST 데이타의 경우에는 인식률 99.11%이었다. 이 해들은 모두 Multistage_HGA가 찾았다. 이는 d가 크다고 반드시 좋은 성능을 보이지는 않는다는 사실을 알려준다. 이 사실은 논문[Zhou02]의 주장과 일치한다.

5.5 토 론

지금까지 기존 분류기 앙상블 선택 방법들과 비교 실험을 통해 다음과 같은 결론을 도출하였다.

1. 우수한 분류기만 사용하는 경우 한계가 있었다. Ranking_Simple와 SGA_AfterRanking 방법에서는 우수한 분류기만을 분류기 앙상블로 사용하였지만 낮은 성능을 보여 분류기의 다양성을 고려해야 할 필요성을 보였다.
2. 분류기의 다양성만 고려한 경우에도 한계가 있었다. Clustering_CD 방법에서는 분류기들을 에러에 대한 경향을 기준으로 서로 다른 분류기들을 앙상블에 선택하였다. MNIST 데이타에 대한 Clustering_CD 방법을 사용한 결과를 보면 우수한 해를 찾을 경우와 그렇지 못한 경우에 성능에 변동이 컸다. 이를 통해 정교한 앙상블 탐색의 필요성을 보았다.
3. 제안한 다단계 방법은 기존 방법들에 비해 우수하였다. 결과적으로 CENPARMI 데이타에 대하여 20.0% (1.75% -> 1.40%)의 오류 감소율을 얻었고, MNIST 데이타에 대하여 14.4% (1.04% -> 0.89%)의 오류 감소율을 얻었다.
4. Multistage_HGA은 분류기 앙상블 선택 문제에 유용하였다. Multistage_SGA에 대한 오류율을 CENPARMI 데이타에 대하여 15.2%(1.65% -> 1.40%) 감소시켰고, MNIST 데이타에 대하여 6.3%(0.95% -> 0.89%) 감소시켰다.

6. 결 론

이 논문은 분류기 앙상블 선택을 위한 다단계 알고리즘을 제안하였다. 이 알고리즘은 초기 분류기 풀에 충분히 다양한 분류기를 확보한 후, 거친 선택과 정교한 선택을 거쳐 우수한 분류기 앙상블을 찾는 전략을 사용한다. 이 알고리즘은 중요한 두 가지 목적, 즉 분류기의 상호 보완성을 유지하며 효율적인 계산 시간 내에 해를 찾는 목적을 달성하였다. 거친 선택 단계에서는 군집화 방법을 이용하였고, 정교한 탐색은 유전 알고리즘을 사용하였다. 또한 탐색 성능이 개선된 혼합 유전 알고리즘을 제시하였다.

잘 알려진 데이타 집합을 이용하여 기존 분류기 앙상블 선택 방법들과 제안한 방법의 비교 실험을 하였고, 제안한 알고리즘이 우수함을 증명할 수 있었다. 또한 SGA와 HGA의 성능 비교를 통해 HGA의 탐색 능력의 우수성을 보였고, 또한 기존 방법의 성능과 HGA에 의한 성능을 비교 분석하였다.

향후 연구로는 유전 알고리즘의 탐색 성능을 향상시키기 위한 새로운 지역 탐색 연산의 개발과 공유(sharing) 기법의 도입이 있다. 또한 부류-모듈러 기법에 기반한 다중 인식기 결합 방법의 연구도 있다.

참 고 문 헌

[1] [Banfield07] R.E. Banfield, L.O. Hall, K.W. Bowyer, and W.P. Kegelmeyer, "A comparison of decision tree ensemble creation method," *IEEE Tr. Pattern Analysis and Machine Intelligence*, vol.29, no.1, pp.173-180, January 2007.

[2] [Fumera05] G. Fumera and F. Roli, "A theoretical and experimental analysis of linear combiners for multiple classifier systems," *IEEE Tr. Pattern Analysis and Machine Intelligence*, vol.27, no.6, pp.942-956, 2005.

[3] [Giacinto01] G. Giacinto and F Roli, "An approach to the automatic design of multiple classifier systems," *Pattern Recognition Letters*, vol.22, pp.25-33, 2001.

[4] [Granitto05] P.M. Granitto, P.F. Verdes, and H.A. Ceccatto, "Neural network ensembles: evaluation of aggregation algorithms," *Artificial Intelligence*, vol.163, no.2, pp.139-162, 2005.

[5] [Hao03] H. Hao, C.-L. Liu, and H. Sako, "Comparison of genetic algorithm and sequential search methods for classifier subset selection," *Proceedings of ICDAR*, 2003.

[6] [Ho02] Tin Kam Ho, "Multiple classifier combination: lessons and next steps," in *Hybrid Methods in Pattern Recognition*, (Ed. by H. Bubke & A.

Kandel), pp.171-198, World Scientific, 2002.
[7] [Kittler03] J. Kittler and F.M. Alkoot, "Sum versus vote fusion in multiple classifier systems," *IEEE Tr. Pattern Analysis and Machine Intelligence*, vol.25, pp.110-115, 2003.
[8] [Liu03] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: benchmarking of state-of-the-art techniques," *Pattern Recognition*, Vol.36, pp.2271-2285, 2004.
[9] [Oh98] Il-Seok Oh and Ching Y. Suen, "Distance features for neural network-based recognition of handwritten characters," *International Journal on Document Analysis and Recognition*, vol.1, pp.73-88, 1998.
[10] [Oh04] Il-Seok Oh, Jin-Seon Lee, and Byung-Ro Moon, "Hybrid genetic algorithms for feature selection," *IEEE Tr. Pattern Analysis and Machine Intelligence*, vol.26, no.11, pp.1424-1437, 2004.
[11] [Partridge96] D. Partridge and W. B. Yates, "Engineering multiversion neural-net systems," *Neural Computation*, vol.8, pp.869-893, 1996.
[12] [Pedrajas05] N. Garcia-Pedrajas, C. Hervas-Martinez, and D. Ortiz-Boyer, "Cooperative coevolution of artificial neural network ensembles for pattern classification," *IEEE Tr. Evolutionary Computation*, Vol.9, No.3, pp.271-302, June 2005.
[13] [Rodriguez06] J.J. Rodriguez, L.I. Kuncheva, and C.J. Alonso, "Rotation forest: a new classifier ensemble method," *IEEE Tr. Pattern Analysis and Machine Intelligence*, vol.28, no.10, pp.1619-1630, October 2006.
[14] [SVMLIGHT07] http://svmlight.joachims.org, 2007.
[15] [Sharkey00] A.J.C. Sharkey, N.E. Sharkey, U. Gerecke, and G.O. Chandroth, "The test and select approach to ensemble combination," in *Multiple Classifier Systems* (Ed. by J. Kittler and F. Roli), Springer, 2000.
[16] [Sohn07] S.Y. Sohn and H.W. Shin, "Experimental study for the comparison of classifier combination methods," *Pattern Recognition*, Vol.40, pp.33-40, 2007.
[17] [Theodoridis06] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 3rd ed., Academic Press, 2006.
[18] [Ueda00] N. Ueda, "Optimal linear combination of neural networks for improving classification performance," *IEEE Tr. Pattern Analysis and Machine Intelligence*, vol.22, no.2, pp.207-215, 2000.
[19] [Wanas06] N.M. Wanas, R.A. Dara, M.S. Kamel, "Adaptive fusion and co-operative training for classifier ensembles," *Pattern Recognition*, Vol.39, pp.1781-1794, 2006.
[20] [Zhou02] Zhi-Hua Zhou, Jianxin Wu, and Wei Tang, "Ensembling neural networks: many could be better than all," *Artificial Intelligence*, vol.137, pp.239-263, 2002.
[21] [문병로03] 문병로, *유전알고리즘*, 두양사, 2003.
[22] [이진선05] 이진선, 김영원, 오일석, "대용량 분류에서 SVM과 신경망의 성능 비교," 정보처리 학회 논문지, vol.12-B, no.1, pp.25-30, 2005.

김 영 원

2001년 전북대학교 컴퓨터과학과 졸업(학사). 2003년 전북대학교 컴퓨터정보학과 졸업(석사). 2006년 8월 전북대학교 컴퓨터통계정보학과 졸업(박사). 2006년 9월~2007년 6월 전북대학교 BK21 사업단 Post-Doc. 2007년 7월~현재 한국전자통신연구원 우정기술연구센터 연구원. 관심분야는 워터마킹, 문자인식, 유전 알고리즘, 컴퓨터비젼

오 일 석

1984년 서울대학교 컴퓨터공학과 졸업(학사). 1992년 KAIST 전산학과 졸업(석사, 박사). 1992년 9월~현재 전북대학교 전자정보공학부 교수. 2005년 1월~2006년 12월 한국정보과학회 컴퓨터비전 및 패턴인식 연구회 운영위원장. 2006년 9월~현재 한국콘텐츠학회 논문지 편집위원장. 2004년 1월~2004년 12월 한국정보과학회 SA 논문지 편집위원장. 관심분야는 문서영상 처리, 패턴인식, 유전알고리즘의 패턴인식 응용