한국지능시스템학회 논문지 2011, Vol. 21, No. 4, pp. 487-492
http://dx.doi.org/10.5391/JKIIS.2011.21.4.487

차분진화 알고리즘을 이용한 Nearest Prototype Classifier 설계

Design of Nearest Prototype Classifier by using Differential Evolutionary Algorithm

노석범 · 안태천*
Seok-Beom Roh and Tae-Chon Ahn

원광대학교 전자 및 제어 공학부
원광대학교 부설 공업기술개발연구소

요 약

본 논문에서는 가장 단순한 구조를 가진 Nearest Prototype Classifier의 성능 개선을 위해 차분 진화 알고리즘을 적용하여 prototype의 위치를 결정하는 방법을 제안하였다. 차분 진화 알고리즘을 이용하여 prototype의 위치 벡터가 결정이 되며, 차분 진화 알고리즘에 의해 결정된 prototype의 class label을 결정하기 위한 class label 결정 알고리즘도 제안하였다. 제안된 알고리즘의 성능 평가를 위해 기존의 패턴 분류기와 비교 결과를 보인다.

키워드 : Nearest Neighborhood classifier, Nearest Prototype classifier, 차분진화 알고리즘, 벡터 양자화

Abstract

In this paper, we proposed a new design methodology to improve the classification performance of the Nearest Prototype Classifier which is one of the simplest classification algorithm. To optimize the position vectors of the prototypes in the nearest prototype classifier, we use the differential evolutionary algorithm. The optimized position vectors of the prototypes result in the improvement of the classification performance. The new method to determine the class labels of the prototypes, which are defined by the differential evolutionary algorithm, is proposed. In addition, the experimental application covers a comparative analysis including several previously commonly encountered methods.

Key Words : Nearest Neighborhood classifier, Nearest Prototype classifier, Differential Evolutionary Algorithm, Vector Quantization

1. 서 론

최근연구 되고 있는 패턴 분류를 위한 다양한 접근 방법들 중 신경망 분류기 (neural networks classifier)와 통계적 방법에 기반을 둔 분류기 (statistical classifier)에 대한 연구가 활발하게 진행되고 있다 [1, 2]. 신경망 분류기 부분에서는 다층 퍼셉트론 (Multi-Layer Perceptron; MLP), Radial Basis Function Neural Networks (RBFNN)[3, 4], Learning Vector Quantization (LVQ) [5]등이 연구되고 있다.

이와 별개로 통계적 분류기도 Linear Discriminant Function, Quadratic Discriminant Function, Nearest Neighbors Classifier(NNC), Parzen Window 분류기 등의 분야에서 활발하게 연구되어지고 있다. Statistical classifier들 중 NNC와 변형된 NNC는 패턴 분류의 많은 영역에서 우수한 성능을 보이는 것으로 알려져 있다[6]. NNC는 학습 데이터 패턴 전체를 학습 된 prototype들로 저장하고, 새로운 입력이 주어졌을 경우, 주어진 새로운 입력과 가장 가까운 prototype의 정보를 이용하여 새롭게 주어진 입력 패턴을 분류하게 된다. 만약 주어진 학습 데이터의 패턴의 수를 의미가 있는 범위만큼 줄일 수 있다면, NNC는 Nearest Prototype Classifier (NPC)[7, 8]로 불릴 수 있다 [5].

NNC에서 기원한 NPC는 classification을 위해 가장 가깝게 위치한 prototype에 따라 주어진 데이터의 분류가 이루어진다는 점에서 vector quantization 기법과 유사한 특징을 보인다[5].

한편, 대표적인 vector quantization 방법인 LVQ 방법은 패턴 분류기의 성능이 prototype의 개수와 prototype의 초기 위치에 민감하게 의존하기 때문에 우수한 패턴 분류 성능을 보이는 패턴 분류기를 설계하기가 어렵다. 또한 LVQ 알고리즘은 prototype의 위치를 학습하기 위한 학습 알고리즘에 필요한 학습 파라미터에 따라 패턴 분류기의 성능이 영향을 받는다.

본 논문에서는, NPC의 문제를 해결하기 위하여, 합리적

접수일자 : 2011년 5월 22일
완료일자 : 2011년 8월 16일
* : 교신저자
이 논문은 2006년도 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(KRF-2006-353-D00021)

인 구조와 빠른 탐색 속도라는 장점을 가지고 있다고 알려져 있는 최적화 알고리즘 중의 하나인 차분 진화 알고리즘(Differential Evolutionary Algorithm: DEA)[9, 10, 11]을 이용하여 prototype의 위치와 prototype의 class label를 결정함으로써, LVQ에서 prototype의 위치를 학습하기 위한 학습 알고리즘에 필요한 학습 파라미터를 배제한 변형된 NPC 설계 방법을 제안하였다.

끝으로 제안된 분류기의 성능을 검증하기 위해 다양한 machine learning classification 문제에 제안된 분류기를 적용한 분류 성능을 기존 패턴 분류기의 성능과 비교한다.

2. Nearest Prototype Classifier

퍼지 Nearest Prototype Classifier(NPC)는 새로운 데이터 패턴이 주어졌을 경우 주어진 데이터 패턴의 class label를 결정하기 위하여 vector quantization 알고리즘과 동일한 nearest neighbor rule를 사용하는 방법이다. NPC에서는 vector quantization과 마찬가지로 prototype의 입력공간상에서의 위치가 패턴 분류 성능을 결정하는 중요한 요소이다. LVQ 방법에서는 입력공간상에서 prototype들의 최적의 위치를 결정하기 위하여 학습 알고리즘을 사용한다. 입력공간상에서 최적의 위치를 결정하기 위한 학습 알고리즘의 경우, 설계자가 선택해야 할 학습 파라미터들이 존재하며, 학습 알고리즘의 학습 성능은 학습 파라미터의 선택에 많은 영향을 받는다. 또한 학습 알고리즘의 성능은 초기 prototype의 위치에 영향을 받는다. 본 논문에서는 학습 알고리즘을 이용하여 최적의 prototype의 위치를 결정하여야 할 경우, 위에 열거한 학습 파라미터 결정 문제와 prototype의 초기 위치 결정 문제를 해결하기 위하여, 최적화 알고리즘을 사용하였다. 다양한 최적화 알고리즘들 중에서 산술적 연산을 통해 새로운 개체를 생성한다는 이점을 가지고 있으며, 유전자 알고리즘에 비해 간단한 구조와 빠른 탐색 속도를 보이는 진화 연산 알고리즘을 사용하였다.

제안된 알고리즘을 설명하기 위해 사용되어진 수학 기호들은 표 1에 기술된다.

표 1 기본 수학 기호
Table 1 Basic notation used in the study

$(\boldsymbol{x_q}, y_q)$	A query pattern (test pattern) and its label ($\boldsymbol{x_q} \in R^m,\ y_q \in N$)
$\hat{y_q}$	Predicted label of a query pattern
$(\boldsymbol{x_i}, y_i)$	Reference pattern (i.e. training pattern) and its label ($\boldsymbol{x_i} \in R^m,\ y_i \in N,\ i=1, 2, \cdots, n$)
$(\boldsymbol{v_j}, L_j)$	A prototype and its label ($\boldsymbol{v_j} \in R^m,\ L_j \in N,\ j=1, 2, \cdots, p$)

여기서, n은 학습 데이터의 수를 의미하며, p는 prototype의 수를 나타낸다.

2.1 Nearest Prototype Classifier

Nearest Prototype Classifier(NPC)는 가장 간단형태의 패턴 분류기들 중 하나이다 [12].

새로운 데이터 패턴인 $\boldsymbol{x}_q$가 주어졌을 경우, NPC는 (1)을 이용하여 prototype들 중에서 $\boldsymbol{x}_q$와 가장 가까운 거리에 있는 prototype의 class를 $\boldsymbol{x}_q$가 속하게 될 class로 추정한다.

$$\hat{y_q} = L_k, \quad \text{if } k = \underset{i}{\operatorname{argmin}}\, d(\boldsymbol{x_q}, \boldsymbol{v_i}) \tag{1}$$

여기서,$d(\boldsymbol{a}, \boldsymbol{b})$ 는 두 벡터들 사이의 거리를 의미한다.

본 논문에서는 두 벡터들 사이의 거리를 (2)과 같이 정의된 weighted Euclidean distance를 사용하여 계산한다.

$$d(\boldsymbol{x_q}, \boldsymbol{v_i}) = \sqrt{\sum_{l=1}^{m} \frac{(x_{ql} - v_{il})^2}{\sigma_l^2}} \tag{2}$$

여기서, m은 입력변수의 수를 의미하며, σ_l은 입력변수 l의 표준 편차를 의미한다.

NPC의 분류 성능은 prototype의 위치에 영향을 받는다. Prototype을 결정하는 방법은 학습 데이터 중에서 prototype을 선택하는 방법과 새로운 prototype을 만들어 내는 방법으로 나눌 수 있다. Prototype을 학습 데이터에서 선택하는 방법은 유전자 알고리즘 또는 랜덤 탐색과 같은 탐색 알고리즘을 사용하는 방법이 연구되어지고 있으며, 새로운 prototype을 만들어 내는 방법들 중 가장 대표적인 방법이 LVQ 방법이 있다 [12]. 본 논문에서는 prototype의 위치를 결정하기 위하여 학습 데이터들 중에서 prototype을 선택하는 방법을 사용하지 않고 새롭게 prototype의 위치를 결정하기 위하여 최적화 방법을 사용하였다. 다양한 최적화 알고리즘들 중에서 간단한 구조와 빠른 탐색 속도를 가진 차분 진화 연산 알고리즘[13]을 사용하여 prototype의 위치를 결정한다.

3. 차분 진화 연산 알고리즘

차분진화 알고리즘(Differential Evolution Algorithm: DEA)은 Price와 Storn에 의해 벡터 차분(vector differential)을 사용하여 Chebychev 다항곡선의 내삽문제(polynomial fitting problem)를 해결하는 과정에서 개발되었다. 차분진화 알고리즘은 현재 널리 사용되고 있는 통계적 임의 탐색법인 유전 알고리즘(Genetic Algorithm: GA)과 유사한 알고리즘이다[9]. 유전 알고리즘과 차분진화 알고리즘은 초기 개체군을 이루는 개체를 샘플링 알고리즘들의 교배(crossover) 돌연변이(mutation), 선택(selection) 과정을 거쳐 적합도(fitness)가 개선되는 개체들을 추출해 낸다는 공통점이 있다. 그러나 이런 연산 과정에서 GA알고리즘은 개체들의 표현형(phenotype)을 유전형(genotype)으로 바꾸는 코딩(coding)이 필요한 반면, 차분진화 알고리즘은 코딩 과정이 필요 없으며 개체를 벡터로 표현하기 때문에 이들의 산술적 연산을 통해 새로운 개체를 생성한다는 이점이 있다. 또한 유전 알고리즘의 단점이었던 너무 복잡한 구조와 연산 대신 간단하고 합리적인 구조와 빠른 탐색 속도가 장점이다 [13]. 차분진화 알고리즘에서는 초기 개체군을 이루는 모든 개체들을 벡터로 표현하는데, 이들을 파라미터 벡터라고 표현한다. 집단 크기는 진화 과정에서 변하지 않는다.

$$v_{jG} \in R^D \qquad \text{j = 1, 2, …, NP} \tag{3}$$

여기서 NP는 파라미터 벡터의 개수이고, 이 값은 최적화 과정에서 일정하며, 설계자가 문제에 따라 정의할 수 있는

샘플링 개수이다. G는 파라미터 벡터가 몇 번째 세대인가를 표현하며 알고리즘에서는 최적화 횟수(generation)를 의미한다. 파라미터 벡터의 차원은 목적함수를 이루는 설계변수의 개수 D와 같으며, 초기 파라미터는 벡터 가용 영역(feasible region)에서 임의로 선택하되, 일반적으로 균등 확률 분포를 따르도록 한다.

차분 진화는 두 개의 개체 벡터의 차이에 가중치를 곱한 것을 세 번째 개체 벡터에 더해서 교배용 벡터를 발생한다. 발생된 교배 벡터와 교배 대상 벡터가 교배되어 새로운 벡터가 얻어진다. 이 새로운 벡터의 목적함수 값이 교배 대상 벡터의 것 보다 좋으면 이 벡터로 교배 대상 벡터를 대체한다. 이는 유전자 알고리즘에서 사용되는 엘리트전략과 유사한 효과를 거둘 수 있다. 집단에서 선택된 벡터의 거리와 방향 정보를 사용하여 벡터를 랜덤하게 변화시키는 것이 차분 진화로 하여금 뛰어난 수렴 성을 갖도록 해준다.

교배용 벡터는 벡터 $x_{i,G}(i=1,2,...,Popsize)$ 집단에서 랜덤하게 선택된 서로 다른 3개의 벡터로부터 (4)와 같이 만든다.

$$v_{i,G+1}=x_{r1,G}+F\cdot(x_{r2,G}-x_{r3,G}) \tag{4}$$

여기서 r_1,r_2와 r_3은 $[0,Popsize-1]$범위의 서로 다른 정수이고, $0\le F\le 2$, $i\neq r_1,r_2,r_3$. F는 선택된 교배 대상 벡터들의 차이$(x_{r2,G}-x_{r3,G})$을 제어하는 파라미터로 0과 2사이의 값을 갖는다.

$v_{i,G+1}$을 발생하기 위해 선택된 벡터 $x_{r1,G}$는 $v_{i,G}$와 관련이 없는 집단에서 랜덤하게 선택된 개체이다.

차분 변화를 통해 얻어질 벡터는 (4)의 교배용 벡터와 교배 대상 벡터인 $x_{i,G}$를 교배하여 만들어진 것이다. 생성된 벡터의 다양성을 증가시키기 위해 주로 균일 교배과정을 (5)와 같이 이루어진다.

$$x'_{i,G+1}=(x_{1i,G+1},x_{2i,G+1},\cdots,x_{di,G+1}) \tag{5}$$

$$x'_{ji,G+1}=(v_{ji,G+1}*x_{ji,G})\ j=1,\cdots,d \tag{6}$$

$*$는 교배 연산자로 균일 교배가 주로 사용 된다. 균일 교배는 (7)과 같다.

$$x'_{ji,G+1}=\begin{cases}v_{ji,G+1}, & \text{if } rand\le Cr\\ x_{ji,G}, & otherwise\end{cases} \tag{7}$$

여기서, rand는 0과 1사이의 랜덤 변수이고, Cr은 교배율이다.

교배 후 만들어진 개체의 생존 여부는 $x'_{i,G+1}$와 $x_{i,G}$의 성능 비교를 통해 이루어진다.

이러한 과정은 설계자가 정해준 세대횟수만큼 반복하며, 결국 목적 함수 값이 작은 파라미터 벡터들로 구성된 개체군이 만들어진다. 이 벡터 중 가작 작은 목적 함수 값이 전역해가 된다.

차분 진화에서는 실수를 직접 사용하기 때문에 개체 평가 시에 유전자형을 표현형으로 변환할 필요가 없다. 다음 세대를 위한 개체 선정이 적합도 비교에 의해서만 이루어지기 때문에 일반적인 진화알고리즘에서 필요한 선택 메커니즘과 최대 또는 최소화 문제에 따른 적합도 함수로의 변환이 필요 없다. 이는 알고리즘의 처리 속도 개선에 유리하다.

그림 1는 전체적인 진화 연산 알고리즘의 순서도를 보인다.

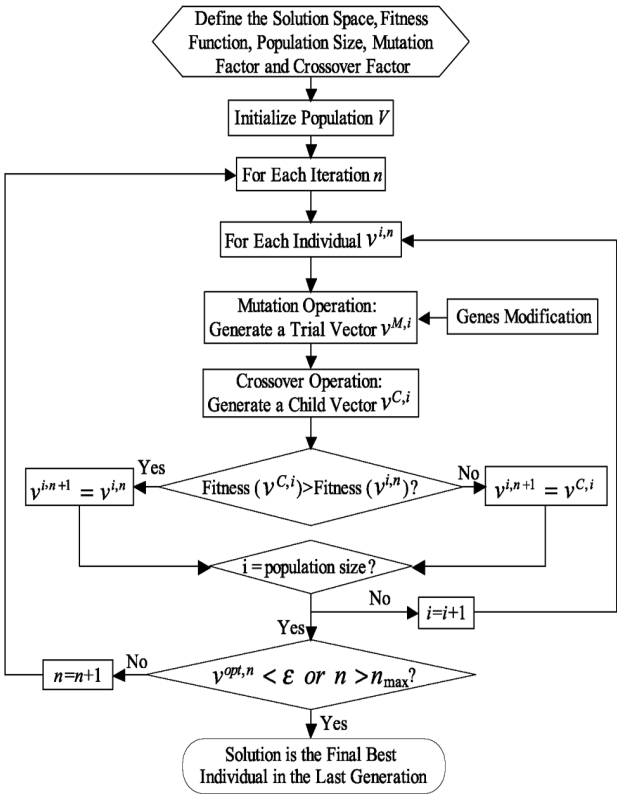

그림 1. 진화 알고리즘의 순서도
Fig. 1 Flowchart of Differential Evolution Algorithm

차분진화의 실행 단계를 정리하면 다음과 같다.

[Step 1] 초기 집단 구성 : 랜덤 값으로 N개의 개체를 초기화. 각 개체는 n개의 목적변수로 구성 (t=0)

[Step 2] 집단내의 모든 개체의 목적함수 평가

[Step 3] 모든 개체(i = 1, ..., N)에 대해 차분 변화를 위한 개체 a_{r1}, a_{r2}와 a_{r3}를 선택하여 교배용 벡터를 만들고 이를 교배 대상 벡터와 교배

[Step 4] 모든 개체의 목적함수 평가

[Step 5] 종료조건을 확인하고 종료조건이 만족되지 않으면 t=t+1로 하고 [Step 3]으로 복귀

4. 차분 진화 연산 알고리즘 기반 Nearest Prototype Classifier

Nearest Prototype Classifier(NPC)의 성능을 개선하기 하기 위하여 앞 장에서 설명한 차분 진화 알고리즘을 사용하여 최적의 prototype을 결정한다.

차분 진화 알고리즘을 이용하여 prototype의 위치를 결정하기 위한 차분 진화 알고리즘의 파라미터 벡터들의 구조는 그림 2와 같다

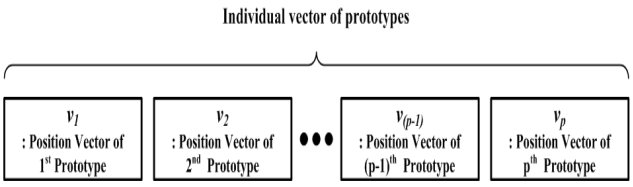

그림 2. 초기 파라미터 벡터의 구조
Fig. 2 Construction of initial parameter vectors

차분 진화 알고리즘의 초기화 단계에서는 모든 벡터에서 각 결정 파라미터는 랜덤으로 할당되고, 돌연변이 과정을 거쳐 재생산과 선택과정을 거친다. 그림 2에 보인 prototype의 최적화를 위한 차분 진화 알고리즘의 초기 파라미터 벡터는 표 1에 보인 prototype의 위치 벡터인 $\{\boldsymbol{v_1}, \boldsymbol{v_2}, \cdots, \boldsymbol{v_{p-1}}, \boldsymbol{v_p}\}$를 표현한다.

차분 진화 알고리즘은 각각의 prototype의 label 값 $\{L_1, L_2, \cdots, L_{p-1}, L_p\}$ 을 결정하지 않는다.

각 prototype의 label을 결정하기 위하여 학습 데이터 패턴을 이용하여 최적의 패턴 분류 성능을 보이는 prototype label을 추정하여야 한다.

NPC의 성능 지수는 (8)과 같이 정의 한다.

$$PI = \left(\frac{1}{n}\sum_{k=1}^{n} g(y_k, \hat{y_k})\right)100 = \left(\frac{1}{n}\sum_{k=1}^{n} g(y_k, \boldsymbol{s_k} \cdot \boldsymbol{L})\right)100 \quad (8)$$

여기서, n은 학습 데이터 패턴의 수를 의미하며, $\boldsymbol{s_k} = [s_{k1}\ s_{k2}\ \cdots\ s_{kp}]$ 는 주어진 임의의 데이터 패턴에 어떤 prototype이 가장 근접해 있는지를 나타내는 selection vector를 의미한다. $\boldsymbol{L} = [L_1\ L_2\ \cdots\ L_p]^T$ 는 각각의 prototype에 할당된 class label들을 의미한다.

또한, $g(a,b) = \begin{cases} 1, & \text{if } a \neq b \\ 0, & \text{if } a = b \end{cases}$, $\hat{y_k} = \boldsymbol{s_k} \cdot L$을 나타낸다.

Selection vector $\boldsymbol{s_k} = [s_{k1}\ s_{k2}\ \cdots\ s_{kp}]$ 의 요소는 (9)를 이용하여 계산한다.

$$s_{kj} = \begin{cases} 1, & \text{if } j = \underset{1 \le i \le p}{\operatorname{argmin}}\ d(\boldsymbol{x_k}, \boldsymbol{v_k}) \\ 0, & \text{if } j \neq \underset{1 \le i \le p}{\operatorname{argmin}}\ d(\boldsymbol{x_k}, \boldsymbol{v_k}) \end{cases} \quad (9)$$

Selection vector $\boldsymbol{s_k} = [s_{k1}\ s_{k2}\ \cdots\ s_{kp}]$ 의 요소인 s_{kj}는 조건 (10)을 만족한다.

$$\sum_{j=1}^{p} s_{kj} = 1 \quad (10)$$

Selection vector를 이용하여 구성된 selection matrix는 (11)과 같다.

$$\boldsymbol{S} = \left[\boldsymbol{s_1^T}\ \ \boldsymbol{s_2^T}\ \ \cdots\ \ \boldsymbol{s_n^T}\right]^T = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1p} \\ s_{21} & s_{22} & \cdots & s_{2p} \\ \vdots & & \cdots & \vdots \\ s_{n1} & s_{n2} & \cdots & s_{np} \end{bmatrix} \quad (11)$$

다른 prototype들에 비해 i번째 prototype에 가까운 데이터 패턴의 수 n_i는 (12)을 이용하여 계산한다.

$$n_i = \sum_{k=1}^{n} s_{ki} \quad (12)$$

다른 prototype들에 비해 i번째 prototype에 가까운 데이터 패턴들 중에서 데이터 패턴의 label이 j번째 class에 속한 데이터 패턴의 수를 n_{ij}로 정의 한다. i번째 prototype의 class label이 j가 될 가능성은 (13)과 같이 구할 수 있다.

$$Poss(L_i = j) = \frac{n_{ij}}{n_i} = \frac{n_{ij}}{\sum_{l=1}^{c} n_{il}} \quad (13)$$

i번째 prototype의 class label은 c개의 class label들이 될 가능성 중에서 가장 높은 가능성을 보이는 class label로 설정되며, (14)를 이용하여 결정한다.

$$L_i = \underset{1 \le j \le c}{\operatorname{argmax}}\ \frac{n_{ij}}{n_i} = \underset{1 \le j \le c}{\operatorname{argmax}}\ \frac{n_{ij}}{\sum_{l=1}^{c} n_{il}} \quad (14)$$

진화 알고리즘의 개체 벡터들의 성능 평가를 위한 적합도 함수는 (8)과 같은 학습데이터 패턴 오인식률을 이용한다.

차분진화 알고리즘을 이용한 NPC의 설계를 위한 알고리즘은 아래와 같다.

[Step 1] 초기 집단 구성 :
- 랜덤 값으로 N개의 개체를 초기화. 각 개체는 p개의 초기 prototype vector 구성

[Step 2] 집단내의 모든 개체의 목적함수 평가

[Sub-step 2.1] 개체 벡터들을 이용하여 prototype의 위치 $\{\boldsymbol{v_1}, \boldsymbol{v_2}, \cdots, \boldsymbol{v_{p-1}}, \boldsymbol{v_p}\}$ 결정

[Sub-step 2.2] $\{\boldsymbol{v_1}, \boldsymbol{v_2}, \cdots, \boldsymbol{v_{p-1}}, \boldsymbol{v_p}\}$ 을 (9)에 적용하여 Selection Matrix를 (11)과 같이 구성

[Sub-step 2.3] Selection matrix를 기반으로 (13)과 (14)를 이용하여 prototype의 class label $\boldsymbol{L} = [L_1\ L_2\ \cdots\ L_p]^T$ 결정

[Sub-step 2.4] $\{\boldsymbol{v_1}, \boldsymbol{v_2}, \cdots, \boldsymbol{v_{p-1}}, \boldsymbol{v_p}\}$, $\boldsymbol{s_k} = [s_{k1}\ s_{k2}\ \cdots\ s_{kp}]$, $\boldsymbol{L} = [L_1\ L_2\ \cdots\ L_p]^T$ 을 (8)에 적용하여 개체의 적합도 계산

[Step 3] 모든 개체(i = 1, ..., N)에 대해 차분 변화를 위한 개체 a_{r1}, a_{r2}와 a_{r3}를 선택하여 교배용 벡터를 만들고 이를 교배 대상 벡터와 교배

[Step 4] 모든 개체의 목적함수 평가

[Step 5] 종료조건을 확인하고 종료조건이 만족되지 않으면 t=t+1로 하고 [Step 3]으로 복귀

5. 실험 및 결과 고찰

본 연구에서는 제안된 분류기의 성능을 평가하기 위하여 제안된 분류기를 기계 학습 데이터 집합 (machine learning dataset)에 적용하여 분류기로서의 성능을 평가 및 분석한다.

기계 학습 데이터 집합은 UCI machine learning repository로부터 획득한 5가지의 데이터 집합들을 이용하여 기존 논문에서 제안된 분류기 기법들과 비교 평가한다.

제안된 분류기의 성능 평가를 위하여 10 fold cross-validation 기법을 10회 적용하여 실험을 수행하게 된다. 제안된 제어기의 분류 성능 평가를 위한 성능 평가 지수는 식 (15)와 같이 패턴 분류기의 오분류(misclassification rate)

비율로 한다.

$$\text{오분류율} = \frac{\sum_{i=1}^{N} g(y_i - \hat{y_i})}{N} \times 100 \quad (15)$$

여기서, N은 데이터의 수를 의미한다.

$$g(y_i - \hat{y_i}) = \begin{cases} 0, & y_i \neq \hat{y_i} \\ 1, & y_i = \hat{y_i} \end{cases} \quad (16)$$

제안된 분류기를 설계할 때 필요한 파라미터들을 표 2에 나타내었다.

표 2. 제안된 분류기의 설계 파라미터
Table 2. Design parameters of the proposed classifier

Number of prototypes (p)	2, 5, 10 , 15, 20
Number of Individuals	100
Number of Generation	50
Crossover rate	1.0

제안된 분류기의 패턴 분류 성능을 비교 평가하기 위해 기존 분류기의 benchmark data로 이용되는 기계 학습 데이터 집합에 적용하였다. 다양한 종류의 기계 학습 데이터 집합들 중에5개의 기계 학습 데이터 집합을 선정하여 실험을 수행 하였다. 선정된 데이터 집합에 대한 일반적인 설명을 표 3에 나열하였다.

표 3. 실험에 사용된 machine learning dataset
Table 3. Machine learning dataset used in the experiment

Dataset	Number of features	Number of patterns
Balance	4	625
Bupa	6	345
Iris	3	150
Pima	8	768
Thyroid(new)	5	215

표 4는 5개의 기계 학습 데이터 집합의 경우에 제안된 분류기와 다른 잘 알려진 분류기들과의 성능을 비교한 것이다.

표 4에 나타난 바와 같이 제안된 패턴 분류기는 Bupa, Iris, Pima 데이터 집합에서는 가장 우수한 패턴 분류 성능을 보이며, Balance 데이터 집합에서는 Naive Bayes 패턴 분류기와 거의 유사한 수준의 분류 성능을 보인다.

마지막으로 Thyroid 데이터 집합의 경우, 제안된 패턴 분류기가 AMPSO, IBK 1, Naive Bayes, RBFNN에 비해 좋지 않은 패턴 분류 성능을 보이지만, 제시된 대부분의 데이터 집합에서는 우수한 성능을 보인다.

표 4. 제안된 분류기와 다른 분류기와의 분류 성능 비교
Table 4. Comparison of the classification error of the proposed classifier and other methods

	Balance	Bupa	Iris	Pima	Thyroid
Proposed Classifier	**89.84 ±0.31**	**73.33± 1.53***	**97.73 ±0.90***	**79.21 ±0.50***	95.16 ±0.80
AMPSO [14]	85.64	65.25	96.89	75.05	96.28
Pittsburgh PSO [14]	62.79	65.13	90.89	74.54	88.93
IBK 1(K=1) [14]	82.45	62.90	95.40	70.44	**97.21**
IBK (K=3) [14]	80.26	63.16	95.20	73.88	93.36
LVQ [14]	85.10	62.18	95.06	74.26	91.03
ENPC [14]	87.00	64.15	95.07	73.54	94.81
J48 (Trees) [14]	77.82	66.01	94.73	74.48	92.06
PART (Rules) [14]	83.17	63.08	94.20	74.18	93.92
Naive Bayes [14]	**90.53***	55.63	95.33	75.69	96.75
SMO (SVM) [14]	87.62	57.95	96.27	76.63	89.74
RBFNN [14]	86.25	65.09	96.00	74.37	96.41
GAssist [14]	81.81	63.21	94.13	73.82	92.04
Fuzzy Rule Extraction [14]	78.22	58.41	94.06	68.47	85.14

* : t-test를 이용하여 검증한 우수한 패턴 분류기

6. 결 론

본 논문에서는 가장 단순한 구조를 가진 Nearest Prototype Classifier(NPC)의 성능개선을 위해 차분 진화 알고리즘을 이용하여 prototype의 위치를 결정하고, 정의된 prototype의 class label을 결정하기 위한 알고리즘을 제안하였다.

제안된 패턴 분류기 설계 방법에 따라 설계된 NPC의 패턴 분류 성능을 평가하기 위하여 다양한 기계학습 데이터 집합에 적용하였다.

제안된 패턴 분류기를 기계학습 데이터 집합에 적용한 실험 결과에 따르면, 3가지의 기계학습 데이터 집합에서 가장 우수한 패턴 분류 성능을 보이고 있으며, 나머지 2가지의 기계학습 데이터 집합에서도 우수한 패턴 분류 성능을 보이는 것을 알 수 있었다.

참 고 문 헌

[1] A.K. Jain, P. W. Duin, J. Mao, "Statistical pattern recognition: a review," *IEEE trans. pattern Anal. Mach. Intell.* vol. 22, no. 1, pp. 4-37, 2000.

[2] C. L. Liu, H. Sako, "Class-specific feature polynomial classifier for pattern classification and its application to hand written numeral recognition," *Pattern recognition*, vol. 39, pp. 669-681, 2006.

[3] U. Krebel, J. Schurmann, "Pattern classification techniques based on function approximation," *Handbook of character recognition and document image analysis, World Scientific*, Singapore, pp. 49-78, 1997.

[4] J Shurmann, *Pattern classification: a unified view of statistical and neural approaches*, Wiley Interscience, New York, 1996.

[5] F. Fernandez, R. Isasi, "Evolutionary Design of Nearest Prototype Classifiers," *Journal of Heuristics*, vol. 10, pp. 431-454, 2004.

[6] W. Lam, C. K. Keung, C. X. Ling, "Learning good prototypes for classification using filtering and abstraction of instances," *Pattern Recognition*, vol. 35, pp. 1491-1506, 2002.

[7] J. C. Bezdek, L. I. Kuncheva, "Nearest Neighbor classifier designs: An Experimental Study," *International Journal of Intelligent Systems*, vol. 16, pp. 1445-1473, 2001.

[8] Ludmila I. Kuncheva, James C. Bezdek, "A Fuzzy Generalized Nearest Prototype Classifier," *IFSA 97 Prague Proceedings Ⅲ*, pp. 217-222, 1997.

[9] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag*, Berlin Heidelberg, 1996.

[10] 황희수, "적응성 있는 차분 진화에 의한 함수최적화와 이벤트 클러스터링," *한국 지능시스템학회 논문지*, 제12권, 5호, pp. 451-461, 2002.

[11] 조세희, 정대형, 오성권, "차분진화 알고리즘을 이용한 FCM 기반 퍼지시스템의 최적설계에 관한 연구," *한국지능시스템학회 2011년도 춘계학술대회 학술발표논문집*, 제21권, 1호, pp. 131-132, 2011.

[12] L. I. Kuncheva, J. C. Bezdek, "Nearest Prototype Classification: Clustering, Genetic Algorithm, or Random Search?," *IEEE Trans. On Systems, Man, and Cybernectics-Part C*, vol. 28, no. 1, pp. 160-164, 1998.

[13] R. Storn, K. V. Price, "Differential Evolution-a fast and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341-359, 1997.

[14] A. Cervantes, I. M. Galvan, P. Isasi, "AMPSO: A New Particle Swarm Method for Nearest Neighborhood Classification," *IEEE Trans. On Systems, Man, and Cybernectics-Part B*, vol. 39, no. 5, pp. 1082-1091, 2009.

저 자 소 개

노석범 (Seok-Beom Roh)

1994년 : 원광대 제어계측공학과 졸업
1996년 : 동 대학원 컴퓨터공학과 석사
2006년 : 동 대학원 제어계측공학과 박사

관심분야 : 퍼지 이론, 신경회로망, Bio-inspired optimization algorithm, Pattern Recognition
E-mail : nado@wku.ac.kr

안태천(Tae-Chon Ahn)

1980년 : 연세대학교 전기공학과 석사
1986년 : 연세대학교 전기공학과 박사
1981년～현재 : 원광대학교 전기전자 및 정보공학부 교수

관심분야 : 지능형 계측 시스템, 지능형 패턴 인식, 디지털 제어 시스템.
E-mail : tcahn@wku.ac.kr