JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 18, No. 6 : 1283~1293 Jun. 2014

확장형 데이터 표현을 이용하는 이진트리의 룰 개선

전해숙[1] · 이원돈[2*]

Refining Rules of Decision Tree Using Extended Data Expression

Hae Sook Jeon[1] · Won Don Lee[2*]

[1]IT Convergence Technology Research Lab., ETRI, Daejeon 305-700, Korea
[2]*Department of Computer Science, Chungnam National University, Daejeon 305-764, Korea

요 약

유비쿼터스 환경에서 데이터는 빠르게 변하고 새로운 데이터는 시간이 경과함에 따라서 출현한다. 그리고 때로, 메모리 공간이 충분하지 않다면, 모든 과거의 데이터를 잃을 수 있다. 그러므로, 과거의 모든 데이터를 잃지 않도록 또는 데이터를 처리하기 위해서 룰을 만들고 새로운 데이터와 결합하는 문제를 해결할 필요가 있다 . 이진트리를 만들고 룰을 추출할 때, 각 룰의 중요도는 일반적으로 리프의 클래스의 총 개수로 정해진다. 주어진 데이터에 맞는 최소한의 유한한 상태 억셉터를 찾기 위한 계산 문제는 NP 하드 문제이다. 추출된 룰은 정확하지 않고 정보의 유실이 있다고 가정된다. 이러한 전제조건 때문에, 본 논문은 룰을 개선하기 위한 새로운 접근을 제시한다. 이것은 이전 지식 또는 데이터로 된 룰의 중요도를 제어하는 것이다. 룰 개선을 할 때, 본 논문은 다수와 소수 특성을 이용하는 푸루닝 방법을 사용하여 다양한 룰을 만들고 룰의 각각의 중요도를 제어하고 성능의 변화를 관찰한다. 본 본문에서 고정된 중요도를 갖는 확장된 데이터 표현을 갖는 이진트리 분류기가 사용되었다. 시험 결과는 룰 개선을 위한 새로운 정책을 이용해서 수행한 성능이 더 좋을 수 있음을 보여준다.

ABSTRACT

In ubiquitous environment, data are changing rapidly and new data is coming as times passes. And sometimes all of the past data will be lost if there is not sufficient space in memory. Therefore, there is a need to make rules and combine it with new data not to lose all the past data or to deal with large amounts of data. In making decision trees and extracting rules, the weight of each of rules is generally determined by the total number of the class at leaf. The computational problem of finding a minimum finite state acceptor compatible with given data is NP-hard. We assume that rules extracted are not correct and may have the loss of some information. Because of this precondition. this paper presents a new approach for refining rules. It controls their weight of rules of previous knowledge or data. In solving rule refinement, this paper tries to make a variety of rules with pruning method with majority and minority properties, control weight of each of rules and observe the change of performances. In this paper, the decision tree classifier with extended data expression having static weight is used for this proposed study. Experiments show that performances conducted with a new policy of refining rules may get better.

키워드 : 룰 개선, 이진트리, 중요도, 푸르닝, 확장 데이터

Key word : rule refinement, decision tree, weight, pruning, extended data expression

접수일자 : 2014. 04. 25 심사완료일자 : 2014. 05. 15 게재확정일자 : 2014. 05. 29

* **Corresponding Author** Won Don Lee(E-mail:wlee@cnu.ac.kr, Tel:+82-42-821-5448)
Department of Computer Science and Engineering, Chungnam National University, Deajeon 305-764, Korea

Open Access http://dx.doi.org/10.6109/jkiice.2014.18.6.1283 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

전문가와의 상호작용을 통한 지식을 얻는 과정은 일련의 집중적이고 체계적인 인터뷰를 일반적으로, 긴 시간동안 하는 것으로 구성 되고 이런 과정에서 만나게 되는 중요한 문제는 전문가가 날마다 자기 일을 하는데 그 지식을 사용할 수는 있지만, 지식을 기계적인 표현과 응용을 위한 형식으로 구체적으로 요약하거나 일반화 할 수 없는 문제점이 있다[1]. 전문가 시스템은 좋은 성능을 얻기 위하여 많은 양의 지식을 요구하지만 그 지식의 인지 과정은 느리고 비싸고 전문가와 인터뷰를 하고 그들의 지식을 수집하는 훈련된 엔지니어의 부족은 지식 인지의 또 다른 문제이다[1]. 앞에서 언급한 문제들은 전문가 시스템에 대한 기술 초기 시대의 어려움이었을 뿐만 아니라 오늘날 여전히 많은 문제로 인식되고 있다. 지식 인지 (특별히, 기계 학습 분야)는 전문가 시스템 연구에 대한 관심의 주요한 영역이 되어오고 있다[1]. 지식 인지의 대체 방안은 예제로부터, 지식을 배우고 또는 추론할 수 있다는 점에 있다. 전문가가 지식을 표현하기가 어려울 때, 지식을 요약하고 표현하기 위하여 사례 연구(case study)를 문서화하는 것은 상대적으로 매우 쉽다. 또한, 지식 자체는 데이터 구조로 표현이 된 예제로부터 추론될 수 있다. 실질적인 중요도가 있는 일에 대한 인공 지능의 대부분의 응용은 인간 전문가가 사용하는 지식에 대한 모델을 구축하는 것을 기반으로 한다. 몇 가지 경우, 전문가가 수행하는 일은 분류로 간주될 수 있다. 즉, 무엇인가를 그들의 특성에 의해 결정되는 범주나 클래스에 할당하는 것이다. 많은 개발들이 이런 지식 인지의 방법이 전체적으로 가능하다는 것을 증명해오고 있다. 잘 알려진 귀납적 추론 알고리즘은 다음과 같다: CLS, ID3, ID4, CN2, BCT, C4.5, AQ and RULES[1].

최근에, 데이터가 빠르게 변하고 있고 새로운 데이터가 끊임없이 시간이 경과함에 따라서 들어온다. 사람들이 원하는 유용한 정보를 얻기가 어렵다. 우리는 유용한 정보를 가능한 한 빠르게 사용자에게 제공해야 하므로, 많은 데이터에서 정보를 추론할 필요가 있다. 이러한 이유 때문에, 분류 알고리즘이 기계 학습과 데이터 마이닝 분야에서 주요한 역할을 하고 있다. 그리고, 작은 시스템이나 많은 데이터를 처리해야 하는 시스템의 경우에 초기에 수집된 또는 받은 데이터를 이용하여 룰을 만들어야 하는 요구가 있다. 이것이 일종의 룰 개선 문제이다. 첫 번째 데이터, 즉, 기존 데이터가 룰로 표현이 되고 두 번째 데이터, 즉, 새롭게 도착한 데이터가 룰과 결합이 될 때, 기존의 존재하는 룰을 체계적으로 개선할 방안이 고안되어야 한다. 이런 복잡한 문제를 처리하기 위한 시도가 있어왔다. 예를 들어, AQ11은 기존 룰과 새로운 데이터 사이의 불일치 문제를 처리했다[2]. 반면에 ID3는 정확하지 않게 분류된 데이터를 이용하는 이진트리를 수정하는 문제를 해결했다[3]. 그러나, 알고리즘의 개발자가 룰과 직접 새로운 데이터를 처리하는 것을 인식할 수 있는 그런 방법은 지금까지 없었다. 기존 룰과 새로운 데이터가 함께 결합됨에 따라서 완전성, 모순과 불일치를 일으키는 원인이 되는 것이 그들 사이에 어떤 상호작용이 있을 수 있다. 이런 문제를 처리하고 기존의 룰을 개선하기 위해서 본 논문에서는 이진트리 분류기 유추(UChoo)를 사용 한다[4-6].

이진트리를 만들고 룰을 추출할 때 룰의 중요도는 트리 리프에 있는 클래스의 수에 의해 결정이 된다. 그러나 Gold는 주어진 데이터에 양립할 수 있는 최소한의 유한한 상태 억셉터를 발견하는 것이 NP 하드(Non-deterministic Polynomial-time hard) 문제임을 보였다[7]. 주어진 +와 - 예제에 양립할 수 있는 가장 작은 크기의 정규적인 표현을 찾는 계산 문제가 NP 하드 문제이다[7]. 그러므로, 예제로부터 트리를 만들고 추출된 룰은 정확하지 않고 몇몇 정보의 유실이 있을 수 있다. 룰 기반 시스템에서 룰을 개선하는 것은 매우 중요한 일이다. 입력 데이터가 룰과 일치하지 않는다면? 어떻게 룰을 바꿀 수 있을까? 정확하지 않은 룰을 바꾸는 문제는 룰 기반 시스템의 문제로 제기되어 왔다[7]. 이런 종류의 문제 때문에, 본 논문은 룰을 개선하기 위한 새로운 방법을 제안한다. 이진트리를 만들고 룰을 추출할 때, 각 룰의 중요도는 일반적으로, 리프 클래스의 총 수로 정해진다. 본 논문은 룰의 중요도가 리프 클래스의 총 수가 되는 것은 룰을 만드는 과정에서 사라진 정보를 포함하는 중요도의 수로 판단하고 룰의 중요도를 조정하는 알고리즘을 제안한다. 룰을 만들 때, 다양한 시험을 위하여 다수와 소수 특성을 이용하는 푸루닝 방법 (PMM; Pruning Method With Majority and Minority Property)을 이용한다[8].

본 논문의 구성은 다음과 같다. II장에서 우선, C4.5

와 다른 확장형 데이터 표현, 두 번째로, C4.5에서 사용된 것과 비슷한 엔트로피 수식을 이용하는 유추(UChoo) 알고리즘, 그리고 푸루닝 방법 PMM이 기술된다 [4-6, 8-11]. 다음에 룰의 중요도를 개선하기 위한 방법이 III장에 제안된다. IV장은 룰 개선을 위한 새로운 접근에 의한 성능을 보이는 실험 결과가 제공된다. 연구에 대한 결론이 V장에 있다.

Ⅱ. 확장형 데이터 표현을 갖는 분류기

2.1. 확장형 데이터 표현

일반적인 표현을 갖는 데이터의 간단한 예제를 살펴보자. 표 1은 트레이닝 데이터로 구성이 되어 있고 데이터는 3개의 속성과 클래스로 설명된다. 첫 번째와 세 번째 속성은 불연속적이고 반면에 두 번째는 연속적인 속성으로 값의 범위가 60에서 80이다. “Outlook” 속성은 세 개의 결과 즉, Sunny, Overcast 그리고 Rain 이다. “Windy?” 속성은 True 그리고 False 2개의 결과를 갖는다. 클래스는 Play와 Don’t Play 2개의 결과를 갖는다.

표 1. 트레이닝 데이터 세트
Table. 1 Training data set

Outlook	Temp(^{O}F)	Windy?	Class
Sunny	70	False	Don’t Play
Sunny	60	True	Play
Overcast	80	False	Play
Rain	60	True	Don’t Play
Rain	70	False	Play
Rain	80	True	Don’t Play

그림 1는 표 1의 트레이닝 데이터 세트의 수정된 표현을 보여준다. 확장된 데이터 표현이라는 것은 데이터의 각 속성을 나열하는 것이고 간단히 0 또는 1을 넣는 것이다. 표 1에 있는 각각의 데이터는 그림 1의 이벤트 번호 1에서 6으로 변환된다. 이 확장된 표현은 두 가지 주요한 특징이 있다. 첫째는 각 속성 결과에 대해 분포를 갖는다. 결과 값을 더하면 1이 된다. 즉, 각 속성의 결과는 0~1의 확률 값으로 채워진다. 두 번째, 이벤트는 중요도 값을 갖는다. 이것은 이벤트가 얼마나 중요성을 갖는지를 보여준다. 어떤 지식은 특정 필드에서 전문가에 의해 기술될 수 있다. 예를 들어, 전문가가 Sunny이고 온도가 70이고 Play 할 확률이 2/3이고 “Windy?”는 고려하지 않는다고 하자. 게다가 전문가는 본인이 설명한 지식이 데이터 세트 안의 개별적인 데이터와 비교했을 때, 30 정도의 중요도를 갖는다고 설명한다고 하자. 유추 알고리즘은 쉽게 전문가의 지식을 추가적인 정보처리 없이 그림 1의 이벤트 7로써 쉽게 표현할 수 있다. 그림 1의 각 열은 중요도 값을 갖는다. 이것은 이벤트가 얼마나 중요한지를 보여주는 것이다.

각 이벤트는 중요도를 갖기 때문에 하나의 이벤트는 그림 1의 중요도 1을 갖는 이벤트로 재정의 되었다. 한 개의 인스턴스는 중요도 1을 갖는 것이 인스턴스이다. 그러므로, 이벤트의 수가 전체 인스턴스의 수와 같지 않을 수 있다. 예를 들어, 그림 1에서, 전체 인스턴스의 수는 36이고 반면에 이벤트의 수는 7이다.
유추(UChoo) 알고리즘은 이 확장된 데이터 표현을 이용하여 다양한 데이터로부터 분류기를 만든다.

Event#	Weight	Outlook			Temp(°F)			Windy?		Class	
		Sunny	Overcast	Rain	60	70	80	True	False	Play	Don't Play
1	1	1	0	0	0	1	0	0	1	0	1
2	1	1	0	0	1	0	0	1	0	1	0
3	1	0	1	0	0	0	1	0	1	1	0
4	1	0	0	1	1	0	0	1	0	0	1
5	1	0	0	1	0	1	0	0	1	1	0
6	1	0	0	1	0	0	1	1	0	0	1
7	30	1	0	0	0	1	0	1/2	1/2	2/3	1/3

그림 1. 표1의 예제를 이용한 확장형 데이터 표현과 한 개의 전문가 지식
Fig. 1 Extended data expression using examples of the Table 1 and the one knowledge of an expert

2.2. 유추(UChoo)

유추(UChoo) 알고리즘은 C4.5것과 비슷한 엔트로피를 사용한다 [4-6, 9-10]. 그러나, 이것은 확장된 데이터 표현을 적용함으로써 수정되었다. 유추(UChoo) 는 자연스럽게 C4.5처럼 관찰된 데이터로부터 분류기를 생성하고 추가적이 개입 없이, 또한 이전의 데이터 세트로부터 만들어진 룰과 뒤에 관찰된 데이터를 결합함으로써 분류기를 또한 만든다. 게다가 유추(UChoo) 는

확장된 데이터 형태를 갖는 전문가가 표현하는 구체적인 데이터도 처리한다.

C4.5의 엔트로피 측정 수식은 다음과 같다.

$$Gain\_ratio\ (A) = Gain\ (A) / Split\_info\ (A) \quad (1)$$

Gain(A)은 속성 A에 따라서 데이터 세트 T를 나눔으로써 얻어지는 정보를 측정하는 것이다. Gain(A) 이론은 속성 A를 선택해서 정보 Gain을 최대화 시키는 것을 선택하는 것이다 (이것은 속성 A와 클래스간의 상호간의 정보로써 알려진 것이다). Split_info(A)는 정규화 상수이다. 속성 A의 결과들의 수가 정보 측정에 영향을 주지 않게 하는 것이다. 데이터 세트 T의 전체 인스턴스의 수는 |T|로 표기한다. 반면에 이벤트의 수는 p(T)이다. T_{Aj} 는 T의 부분집합이다. 이것은 속성 A의 결과 j값을 갖는 것이다. $|T_{Aj}|$는 T_{Aj} 부분집합에 있는 인스턴스의 수이다.

$$Gain(A) = info\ (T) - info_A\ (T) \quad (2)$$

$$info(T) = \sum_{i=1}^{k} (freq(C_i, T)/|T|) \cdot \log_2(freq(C_i, T)/|T|), \quad (3)$$

$$info_A(T) = \sum_{j=1}^{n} (|T_{Aj}|/|T|) \cdot info(T_{Aj}), \quad (4)$$

$$info_A(T_{Aj}) = \sum_{j=1}^{k} (freq(C_i, T)/|T_{Aj}|) \cdot \log_2(freq(C_i, T)/|T_{Aj}|) \quad (5)$$

$$Split\_info(A) = \sum_{j=1}^{n} (|T_{Aj}|/|T|) \cdot \log_2(|T_{Aj}|/|T|). \quad (6)$$

A는 속성이고 k는 클래스의 수이다. 그리고 n은 속성의 결과의 수이다.

클래스 맴버 중요도 $C_i(m)$는 m번째 이벤트가 클래스 C_i에 얼마나 속해있는지를 얘기한다. 그리고 다음 조건을 만족해야 한다:

$$\sum_{i=1}^{k} C_i(m) = 1. \quad (7)$$

속성의 결과 맴버 중요도 $O_{Aj}(m)$가 속성 A가 결과 j가 m번째 이벤트에서 얼마나 발생하는지를 나타낸다. 그리고 다음 조건을 만족해야 한다:

$$\sum_{j=1}^{k} O_{Aj}(m) = 1. \quad (8)$$

$freq(C_i, T)$는 C_i 클래스 값을 갖는 T 세트 안에서 인스턴스의 수이고 수식은 다음과 같다:

$$freq(C_i, T) = \sum_{m=1}^{p(T)} Weight(m, T) \cdot C_i(m). \quad (9)$$

$$freq(C_i, T_{Aj}) = \sum_{m=1}^{p(T)} Weight(m, T) \cdot C_i(m) \cdot O_{Aj}(m). \quad (10)$$

$Weight(m, T)$는 T세트에서 m번째 이벤트의 중요도 값이다. 인스턴스는 중요도가 1인 이벤트이다. 그러므로, 한 이벤트 중요도가 W이면 이것이 의미하는 것은 속성과 클래스의 동등한 분포값을 갖는 인스턴스가 W개 있다는 것을 의미한다. $|T_{Aj}|$는 T_{Aj} 세트 안에 있는 인스턴스의 수이고 다음과 같다:

$$|T_{Aj}| = \sum_{m=1}^{p(T_{Aj})} Weight(m, T_{Aj}) \cdot O_{Aj}(m). \quad (11)$$

이런 경우에, $O_{Aj}(m)$, T_{Aj}세트 안에 있는 각각 이벤트의 결과 가능성 값과 각각의 m에 해당하는 $Weight(m, T_{Aj})$를 곱하고 그 전체곱셈의 결과를 더하고 $|T_{Aj}|$를 얻는다. 그러므로, 재귀적 트리의 노드에서 트리를 나누기 위한 가장 좋은 속성은 이 새롭게 정의된 값을 갖는 엔트로피 수식을 이용함으로써 가장 좋은 *Gain_ratio*를 갖는 것이다.

2.3. Pruning Method

실험에 사용된 푸루닝 방법은 이진트리를 위한 다수와 소수 특성을 이용하는 푸루닝 방법 (PMM)을 이용한다[8]. 루트 노드를 포함한 푸루닝을 할 노드들은 톱다운 정책으로 검사된다. 노드의 총 클래스 수와 그 노드의 가장 큰 클래스의 수의 비율이 어떤 수용할 범위

보다 크면 그 노드는 다수를 대표하는 클래스로 푸루닝 할 것이다. 이 어떤 수용할 범위가 지역적 견해를 갖는 다수 인자 파라메타이다 (MFL; majority factor with the local viewpoint). 트리 전체적인 관점에서 볼 때, 후보 노드를 대표하는 클래스의 수가 어떤 수용할 범위보다 작으면 소수를 대표하는 클래스로 푸루닝 된다. 이 어떤 수용할 범위가 전체적 관점을 갖는 소수 인자 파라메타라고 한다 (MFG; minority factor with the global viewpoint). 이것이 MFL과 MFG를 갖는 푸루닝 방법(PMM)이다. MFL이 1이면 푸루닝 하지 않는다는 것이다. MFL 값이 1보다 낮아지면, 푸루닝의 기회가 더 커진다. MFG가 0이면 푸루닝 할 것이 없다는 것이다. MFG 값이 0보다 커지면, 푸루닝의 기회가 더 커진다.

Ⅲ. 룰 개선

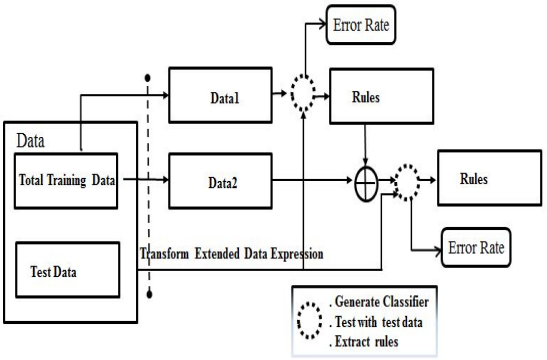

그림 2. 룰 개선 절차
Fig. 2 Procedure of Rule Refinement

룰 개선을 하는 문제는 다음과 같다[4-6]. 이 문제의 데이터가 두 부분으로 분리 된다: 첫 번째 데이터(data1) 그리고 두 번째 데이터(data2). 첫 번째 데이터는 실제 세상에서 조기에 수집되거나 특정 분야의 전문가로 부터주어 진다. 확장된 데이터 표현에서 중요도 필드가 포함된다. 본 논문에서 사용되는 데이터의 각 이벤트의 중요도는 고정 중요도로 불리는 1이 할당된다. 즉, 데이터의 모든 이벤트는 동일한 중요성을 갖는다. 첫 번째 데이터를 이용하는 룰을 만들 때, 이진트리에서 추출되는 룰의 중요도는 리프의 클래스 수를 의미한다. 분류기는 이진트리를 만들고 첫 번째 데이터를 이용하는 룰을 추출한다. 룰의 각각의 중요도가 불필요한 정보를 가졌다고 가정하고 이것을 제어하고자 한다. 분류기는 첫 번째 데이터의 룰과 첫 번째 데이터와 같은 실제 분야에서 수집된 두 번째 데이터로 구성되는 입력을 받는다. 이런 식으로 그림 2와 같이 룰 개선이 이뤄진다. 시험을 통해 얻은 결과는 시험 데이터에 대한 정확도 또는 에러가 된다.

```
Function Name : controlWeight()
Let T be the training data
Let k be the number of rules extracted data1.
Let x be the test record x ∈ T.
Let counter show change of the weight of x.
Let resultFlag notify the result.

Assign TRUE to resultFlag.
for i = 1 to k do
    1. if ((x_i.weight!=1) && (x_i.weight>10))
         res = div(x_i.weight , 2);
         x_i.weight = res.quot; counter++;
    2. if ((x_i.weight!=1) && (x_i.weight<= 10))
         x_i.weight = x_i.weight-1; counter++;
    3. if (counter ==0) resultFlag =FALSE;
end for
return resultFlag.
```

그림 3. 룰의 중요도를 제어하기 위한 알고리즘
Fig. 3 Algorithm for controlling the weight of each of rules

그림 3는 룰의 중요도를 제어하는 절차를 보여준다. 알고리즘은 우리가 생각한 아이디어를 구현하기에 간단하다. 3가지의 시험 케이스가 있다: 첫째는 룰의 모든 중요도가 1이어서 룰의 중요도가 변하지 않는 경우 둘째는, 중요도의 값이 10보다 큰 경우, 계속 2로 나누고 몫을 취한다. 마지막으로 중요도가 한 자리 수인 경우이다, 1이 될 때까지 －1을 한다. 이것은 매우 간단한 알고리즘이고. 다음 장에서 이 제안된 알고리즘을 이용하여 룰의 중요도를 제어하는 시험이 진행된다.

Ⅳ. 실 험

알고리즘은 UCI(University of California Irvine) 기계 저장소에 수집된 데이터로 시험 한다: Zoo, Iris, Sonar, Glass, Ionosphere, Pima Indian Diabets, Car, Splice, Nursery[12]. 데이터들은 잃어버린 값이나 알려지지 않은 값을 포함하지 않는다.

시험의 목적은 룰의 중요도가 3장에서 제안된 알고

리즘에 따라 수행될 때 정확도가 증가되는 경우가 있음을 확인하는 것이다. 우선 각 데이터는 2개의 데이터 세트로 분류 한다: 첫 번째(90%)는 분류기를 훈련하기 위래 사용되고 두 번째 (10%)는 시험하기 위해 사용된다. 이 분류를 10번 반복하는데, 이것이 우리가 사용할 10 cross validation 시험 (10CVT)라고 한다. 여기에서 시험을 위해 첫 번째 데이터가 다시 두 개의 부분집합으로 나누어진다. 첫 번째 (10%) 데이터1(data1) 은 푸루닝 방법인 PMM을 이용해서 룰 개선 문제의 룰을 만들 때 사용한다. 두 번째(90%) 데이터2(data2)는 룰 개선 문제의 데이터로 사용한다. 첫 번째 데이터를 10%만 잡은 이유는 작은 데이터를 가지고 만들어진 룰의 영향력을 동시에 확인하고자 하는 것이다.

본 논문은 전체데이터, 데이터1의 룰과 데이터2로 구성된 결합데이터, 그리고 데이터1의 성능을 비교한다. 결합데이터를 시험할 때, 룰의 중요도 값이 감소함에 따라서 결합데이터의 성능이 전체데이터와 데이터1에 비해서 어떻게 변화하는지를 관찰 할 수 있다. 시험 단계별 최대 성능을 선택하여 정확도 그림에 사용한다.

시험을 할 때, MFL이 1.0이고 MFG가 0.0이면 이것은 푸루닝을 하지 않은 상태이다. 결합데이터의 시험 0(test0)가 룰의 원래의 중요도 값을 이용한 시험이다. 만일 룰의 중요도가 1이 아니면 그림 3에서 제안한 알고리즘을 따라서 룰의 중요도가 1이 될 때까지 감소한다. 시험 단계가 데이터의 정확도 그림들에서 시험0만 있다면, 룰의 모든 중요도가 1인 경우이다.

푸루닝을 하는 개수는 데이터 수마다 다르다. 데이터의 수가 500이하 작은 수이면, 사용된 푸루닝 인자는 다음과 같다: MFL은 1에서 0으로 －0.1씩 감소하고 MFG는 0에서 1로 0.1씩 증가한다. 사용된 MFL의 수는 5이고 MFG의 수는 6이다. 이 푸루닝 인자들은 쌍으로 사용되어야 하므로, 30개의 푸루닝을 적용한다. 작은 데이터에 해당하는 데이터는 Zoo, Iris, Sonar, Glass, Ionosphere 이다. Pima Indian Diabets와 Splice 의 경우는 작은 데이터와 조금 다르다. MFL은 데이터의 수가 작은 경우와 같다. MFG는 0에서 0.009까지 0.0015씩 증가한다. 이 데이터들은 35개의 푸루닝을 한다. Car의 경우는 MFL은 데이터의 수가 작은 경우와 같다. MFG는 다음과 같다:0, 0.001, 0.003, 0.005, 0.007, 0.01. Car 데이터는 30개의 푸루닝을 한다. Nursery는 만개가 넘는 데이터다. MFL은 1에서 0.6까지 0.5씩 감소한다. MFG는 다음과 같다: 0.0001, 0.0003, 0.0005, 0.0007, 0.0009, 0.001. Nursery 데이터는 63개의 프루닝을 한다. 시험 데이터 리스트는 표 2에 나열한다. 훈련 데이터 크기와 시험 데이터 크기를 포함하는 데이터 크기, 속성과 클래스의 수, 데이터1과 데이터2의 수를 포함한다.

표 2. 시험 데이터 구성
Table. 2 Target Test Data Format

데이터	개수 (훈련/시험)	속성 개수	클래스 개수	data1 개수	data2 개수
Zoo	101(90/10)	17	7	9	81
Iris	150(134/15)	4	3	13	120
Sonar	210(188/21)	60	2	18	169
Glass	214(192/22)	9	6	19	172
Ionosphere	351(315/36)	34	2	31	283
Pima Indian Diabets	768(691/77)	8	2	69	621
Car	1728 (1555/173)	6	4	155	1400
Splice	3190 (2871/319)	60	2	287	2583
Nursery	12960 (11664/1296)	8	5	1166	10497

일반적으로, 이 시험을 통해 얻게 될 좋은 성능에 대한 예측은 첫째는 모든 데이터시험에 대한 것이고 둘째는 결합된 데이터의 성능 그리고 마지막이 데이터1의 성능이다. 시험의 성능 즉, 정확도에 영향을 줄 수 있는 것은 첫째, 룰의 중요도를 제어하기 위한 알고리즘의 적용이고 둘째, 푸루닝이 적용되는 것이다. 시험을 통해, 이 두 가지 방법이 시험의 결과 즉, 정확도에 어떠한 영향을 주는지를 확인하고자 한다. 각각의 데이터 시험에 대한 설명은 각각의 데이터별로 기술하고 마지막으로 전체적으로 성능에 대해서 비교를 서술한다.

4.1. Zoo 데이터

데이터1의 수가 9이고 데이터1의 룰의 수가 2이다. 예를 들어, 룰의 각각의 중요도는 5와 4 또는 3과 6이다. 전체데이터의 최대 정확도는 0.9600이고 데이터1의 최대 정확도는 0.5000이다. 룰의 최대 중요도는 7이다. 룰 중요도 변화에 따른 제안된 알고리즘에 따라서 결합데

이터의 시험은 7번 시행 한다:test0~test6. 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.5200이고 시험 6의 정확도는 0.7333이다. 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.8200이고 시험 6의 최대 정확도는 0.9000이다. 그림 4는 Zoo 데이터에 대한 정확도이다. 전체데이터의 성능이 최고이다. 데이터1의 성능이 최저이다. 결합데이터는 시험0에서 가장 최저이고 시험6에서 최대 값을 보여준다. 푸루닝 하지 않은 경우는 21.33%, 푸루닝을 하는 경우는 8% 성능 증가를 보인다.

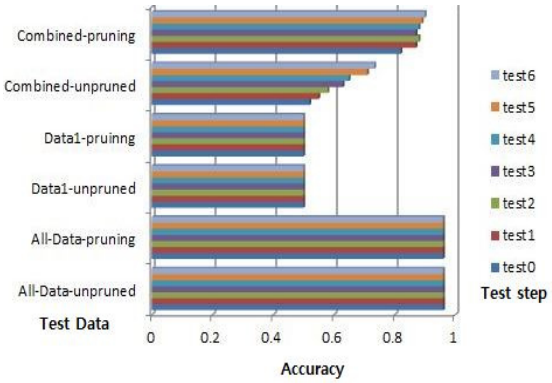

그림 4. Zoo 데이터의 정확도
Fig. 4 Accuracy on Zoo data

4.2. Iris 데이터

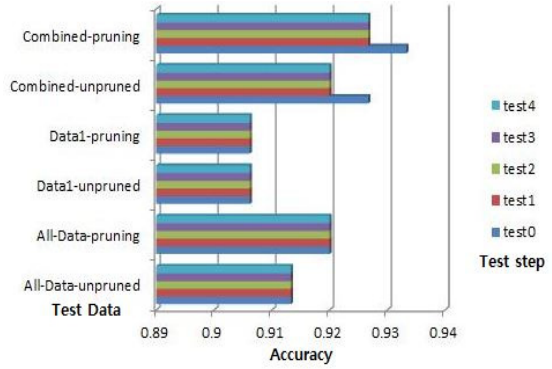

그림 5. Iris 데이터의 정확도
Fig. 5 Accuracy on Iris data

데이터1의 수가 13이고 데이터1의 룰의 수가 3이다. 예를 들어, 룰의 각각의 중요도는 5, 3, 5이다. 전체데이터의 최대 정확도는 0.9200이고 데이터1의 최대 정확도는 0.9062이다. 룰의 최대 중요도는 5이다. 그래서 룰 중요도 변화에 따른 제안된 알고리즘에 따라서 이 데이터의 결합데이터의 시험은 0~4이다:test0~test4. 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.9267이고 시험 4의 정확도는 0.9200이다. 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.9333이고 시험 4의 최대 정확도는 0.9267이다. 그림 5는 Iris 데이터의 정확도이다. 결합데이터의 경우가 전체데이터의 성능보다 높다. 푸루닝 하지 않은 결합데이터는 푸루닝 하지 않은 전체데이터 보다 1.34% 높다. 그리고 푸루닝 한 결합데이터의 성능이 푸루닝한 전체데이터보다 1.33% 높다. 푸루닝한 결합데이터의 성능이 안한 경우보다 0.66% 높다. 결합데이터의 시험0이 제일 성능이 좋고 시험2-4는 시험0보다 성능이 떨어지고는 변하지 않는 상태를 보인다.

4.3. Sonar 데이터

데이터1의 수가 18이고 데이터1의 룰의 수가 4이다. 전체데이터의 최대 정확도는 0.6845이고 데이터1의 최대 정확도는 0.6702이다. 룰의 최대 중요도는 1이다. 그래서 룰 중요도 변화에 따른 제안된 알고리즘에 따라서 이 데이터의 결합데이터의 시험 단계는 시험0이다. 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.6605이고 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.7133이다 . 푸루닝을 한 결합데이터의 성능이 최고 좋고, 다음은 전체데이터이고 그중에 푸루닝한 것, 푸루닝하지 않은 것 그리고 푸루닝한 데이터1 그리고 푸루닝하지 않은 데이터1과 결합데이터의 성능이 최하이다.

4.4. Glass데이터

데이터1의 수가 19이고 데이터1의 룰의 수가 6 또는 12이다. 예를 들어, 룰의 수가 12인 경우, 모든 중요도는 1이다. 그런데, 룰의 수가 6인 경우, 모든 중요도는 1이고 마지막이 2이다. 전체데이터의 최대 정확도는 0.8201이고 데이터1의 최대 정확도는 0.5463이다. 룰의 최대 중요도는 1, 2, 3, 4, 그리고 6이다. 그래서 룰 중요도 변화에 따른 제안된 알고리즘에 따라서 이 데이터의 결합데이터의 시험은 6번이다:test0~test5.이다. 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.6264이고 시험 3의 정확도는 0.6667이다 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.6729이고 시험5의 최대 정확도는 0.6984이다. 이 경우는 시험4가 이 데이터의 최대 정확도는 0.7302를 갖는다. 그림 6의 Glass 데이터는 결합데이터

중 푸루닝 한 것이, 제안된 알고리즘을 따라 진행하면서 성능이 서서히 증가하는데 전체데이터의 푸루닝 하지 않은 성능을 넘어서기까지 한다. 성능이 5.73% 향상되었다. 결합데이터의 푸루닝 하지 않은 경우는 알고리즘에 따른 시험이 진행되면서 조금이나마 성능이 나빠지다가 시험3에서 성능이 거의 4.03% 가량 향상이 되었다. 이 시험이 중간에 끊어지는 것은 10개의 파일과 푸루닝이 적용되면서 생기는 각각의 룰 파일이 갖는 중요도의 다양성 때문이다.

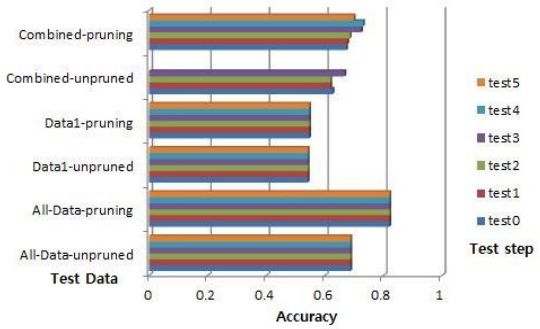

그림 6. Glass 데이터의 정확도
Fig. 6 Accuracy on Glass data

4.5. Ionosphere 데이터

데이터1의 수가 31이고 데이터1의 룰의 수가 3이다. 예를 들어, 룰의 모든 중요도는 1이다. 전체데이터의 최대 정확도는 0.9261이고 데이터1의 최대 정확도는 0.8233이다. 룰의 최대 중요도는 1이다. 그래서 룰 중요도 변화에 따른 제안된 알고리즘에 따라서 이 데이터의 결합데이터의 시험 단계는 시험0이다. 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.8948이고 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.9090이다 . 성능이 좋은 순서가 전체데이터의 경우 중 푸루닝한 것과 푸루닝 하지 않은 것 그리고 결합데이터의 푸루닝한 것과 푸루닝 하지 않은 것 그리고, 데이터1이 마지막이다.

4.6. Pima Indian Diabets 데이터

데이터1의 수가 31이다. 시험할 10개의 파일이 있다. 그 중, 푸루닝이 적용된 파일 0의 경우에 룰 넘버가 11, 9 그리고 8이다. 이 경우, 룰의 최대 중요도는 8과 1이다. 전체 데이터1의 룰 수는 다양하다: 15, 14, 13, 12, 11, 9, 8, 7, 6, 5 그리고 3이다. 룰의 최대 중요도는 10, 9, 8, 3 그리고 1이다. 그래서 룰 중요도 변화에 따른 제안된 알고리즘에 따라서 이 데이터의 결합데이터의 시험 시험은 10번이다:test0~test9. 전체데이터의 최대 정확도는 0.7471이고 데이터1의 최대 정확도는 0.6912이다. 결합데이터의 경우, 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.7093이고 시험 9의 정확도는 0.7403이다 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.7405이고 시험9의 최대 정확도는 0.7403이다. 이 경우는 시험1, 시험2가 데이터의 최대 정확도 0.7727를 갖는다. 그림 7의 Pima Indian Diabets 데이터는 결합데이터의 시험 0-2의 경우에만 파일이 10~8개가 사용되고 시험 3-7은 5개의 파일이 시험8에는 3 개의 파일이 그리고 시험9에는 1개의 파일이 사용되었다. 결합데이터의 성능이 시험0보다 시험 1,2가 더 높음을 보인다. 즉, 3.22% 성능이 좋아졌다.

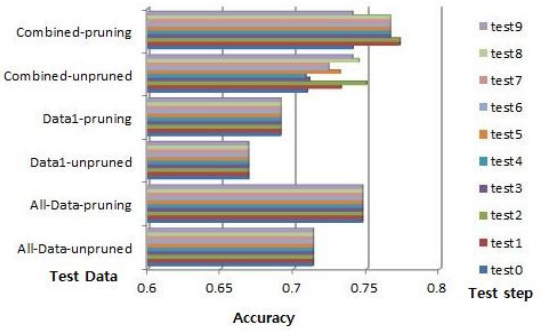

그림 7. Pima Indian Diabets 데이터의 정확도
Fig. 7 Accuracy on Pima Indian Diabets data

4.7. Car 데이터

데이터1의 수가 155이고 데이터1의 룰의 수가 3이다. 예를 들어, 룰의 중요도가 52, 51 그리고 52이다. 룰 중요도 변화에 따른 제안된 알고리즘에 의해 시험은 9단계로 이뤄진다. 전체데이터의 최대 정확도는 0.9769이고 데이터1의 최대 정확도는 0.7002이다. 결합데이터의 경우, 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.7002이고 시험 9의 정확도는 0.7002이다 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.7002이고 시험9의 최대 정확도는 0.9363이다. 그림 8의 Car 데이터는 결합데이터의 시험 0-9의 경우, 꾸준히 성능 증가를 했다. 약 23.61%의 증가를 보이고 있다.

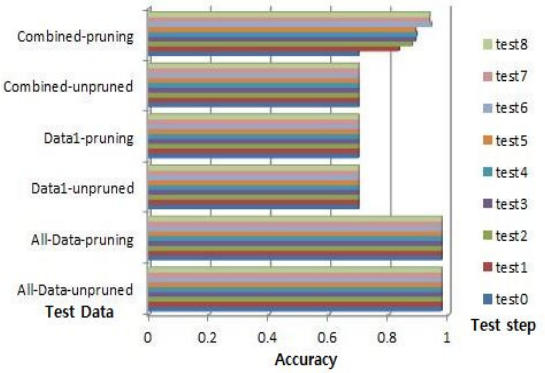

그림 8. Car 데이터의 정확도
Fig. 8 Accuracy on Car data

4.8. Splice 데이터

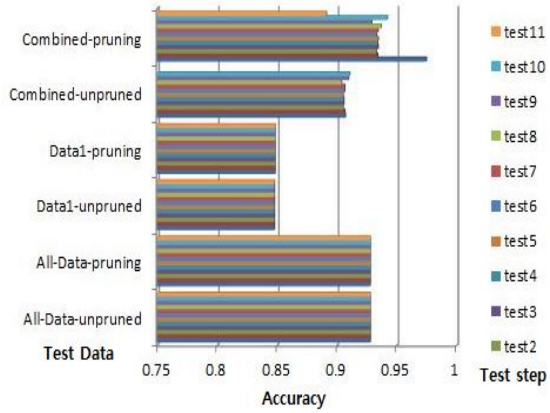

그림 9. Splice 데이터의 정확도
Fig. 9 Accuracy on Splice data

데이터1의 수가 287이고 데이터1의 룰의 수가 21에서 63까지 다양하다: 63, 60, 54, 51, 46, 38, 35, 21. 예를 들어, 이 룰 중에 50개 이상의 중요도가 1이다. 룰 중요도 변화에 따른 제안된 알고리즘에 의해 시험은 12단계로 이뤄진다. 전체데이터의 최대 정확도는 0.9279이고 데이터1의 최대 정확도는 0.8483이다. 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.9069이고 시험 10의 정확도는 0.9107이다. 결합데이터의 경우, 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.9746이고 시험11의 최대 정확도는 0.8911이다.

이 경우는, 시험1이 데이터의 최대 정확도0.9746를 갖는다. 그림 9의 Splice 데이터의 성능이 좋은 순서는 Pima Indian Diabets와 비슷하지만, 결합데이터의 시험 0의 성능이 제일 좋은 것은 Iris 데이터와 유사성을 보인다. 전체데이터 푸루닝 성능보다 결합데이터 푸루닝 성능이 4.67% 높다. 그리고 시험1은 시험0보다 3.26% 성능이 감소했다.

4.9. Nursery 데이터

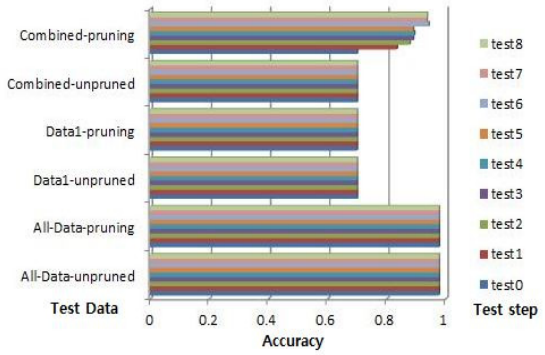

그림 10. Nursery 데이터의 정확도
Fig. 10 Accuracy on Nursery data

데이터1의 수가 1166이고 데이터1의 룰의 수가 184이다. 예를 들어, 이 룰 중에 50개 이상의 중요도가 1이다. 룰 중요도 변화에 따른 제안된 알고리즘에 의해 시험은 12단계로 이뤄진다. 전체데이터의 최대 정확도는 0.9969이고 데이터1의 최대 정확도는 0.9282이다. 결합데이터의 경우, 푸루닝 하지 않은 경우, 시험 0의 정확도는 0.9275이고 시험 11의 정확도는 0.9277이다. 푸루닝을 한 경우, 시험 0의 최대 정확도는 0.9716이고 시험11의 최대 정확도는 0.9812이다. 그림 10의 Nursery 데이터는 결합데이터의 시험 0-7의 경우, 꾸준히 성능 증가를 했다. 약 0.96%의 증가를 보이고 있다.

4.10. 전체적인 성능 비교

본 논문에서 제안한 룰의 중요도를 제어하기 위한 알고리즘과 다양한 시험을 위하여 적용한 푸루닝을 적용한 결과로 각 시험의 결과인 정확도에 어떠한 영향을 미치는지 각 데이터별로 살펴보았다. 룰의 중요도를 제어하기 위한 알고리즘의 적용에 따른 시험 단계에 따라서 정확도가 더 좋아지기를 기대했고 기대에 부응하는 결과를 보여주는 데이터들을 발견하였다.

Data	Accuracy						
	Data1		Data1's rule+Data2			All Data	
			old	new method			
	unpruned	pruned	unpruned	unpruned	pruned	unpruned	pruned
Zoo	0.5000	0.5000	0.5200	0.7333	0.9000	0.9600	0.9600
Iris	0.9062	0.9062	0.9267	**0.9267**	**0.9333**	0.9133	0.9200
Sonar	0.6707	0.6702	0.6605	0.6605	**0.7133**	0.6798	0.6845
Glass	0.5416	0.5463	0.6264	0.6667	0.7238	0.6872	0.8201
Ionosphere	0.8233	0.8233	0.8948	0.8948	0.9090	0.9232	0.9261
Pima Indian Diabets	0.6990	0.6912	0.7093	**0.7499**	**0.7727**	0.7133	0.7471
Car	0.7002	0.7002	0.7002	0.7002	0.9421	0.9769	0.9769
Splice	0.8476	0.8483	0.9069	0.9107	**0.9746**	0.9279	0.9279
Nursery	0.9273	0.9282	0.9275	0.9277	0.9812	0.9969	0.9969

그림 11. 제안된 방법을 이용한 시험 결과
Fig. 11 Experimental Results with proposed method

데이터 Zoo, Glass, Car, 그리고 Nursery인 경우에 푸루닝을 한 결합데이터의 룰의 중요도가 감소함에 따라서, 성능이 비례적으로 증가함을 보인다. 데이터 Sonar, Ionosphere의 경우에는 룰의 중요도가 1이다. 데이터 Ionosphere의 성능은 전체데이터가 첫 번째이고 결합데이터가 두 번째 세 번째는 데이터1이다. Sonar의 경우에는 푸루닝한 결합데이터가 첫 번째이고 푸루닝한 전체데이터가 두 번째이다. Pima Indian Diabets도 Soanr와 비슷하다. Iris와 Splice 데이터의 경우 좀 다르다. 시험0 단계에서 푸루닝한 결합데이터의 성능이 최고이고 룰의 중요도가 감소함에 따라서 성능이 한번 감소하고 그리고 후에는 성능이 변하지 않았다. 본 본문을 생각하면서 예측했던 성능이 좋은 순서는 전체데이터, 결합데이터, 그리고 데이터1이 마지막이다. 그림 11은 각 데이터의 각각의 시험 성능 중에서 최대 정확도를 갖는 것으로 구성했다. 그림 11의 시험 결과를 전체적으로 볼 때, 새로운 알고리즘을 적용한 결과 성능이 좋아졌고 그림 11에서 볼드체로 기록된 성능들 즉, 룰을 이용한 결합데이터의 성능이 모든 성능보다 뛰어난 특별한 경우가 생기는 것을 발견했다.

V. 결 론

확장형 데이터 표현을 이용하는 유추(UChoo) 분류기와 PMM 푸루닝 방법 그리고 본 논문에서 제안하는 룰의 중요도를 개선하기 위한 알고리즘을 룰 개선에 사용하였다. 일반적으로, 전체데이터의 정확도가 결합데이터보다 높다. 시험 결과는 크게 다섯 가지 유형으로 볼 수 있다. 첫째, 제안된 알고리즘이 적용되기 전 최초 시험단계의 성능이 가장 좋고 알고리즘을 적용 후, 성능의 변화가 없는 경우가 있었고 둘째, 알고리즘을 적용을 해 볼 수 없는 룰의 중요도가 모두 1인 경우가 있었다. 셋째, 처음에 예상한대로 전체데이터 성능, 결합데이터 그리고 Data1 성능 순이다. 넷째, 처음 성능이 제일 좋고, 제안된 알고리즘 적용 후, 성능이 떨어지는 경우가 있다. 이 경우는 룰을 만들 때 사용한 중요도의 정보에 사라진 정보가 없는 경우로 추정할 수 있다. 그리고 마지막으로, 시험을 통해서 얻은 결과들을 통해서, 이 새롭게 제안된 룰의 중요도를 개선하기 위한 알고리즘이 룰 개선에 도움이 되고 의미가 있음을 알게 되었다. 즉, 전체시험을 통해서, 제안된 접근 방법에 의한 성능들이 룰들의 각각의 중요도로 클래스의 총 개수를 이용하는 것의 성능보다 더 좋음을 알게 되었다. 뿐만 아니라, 푸루닝을 적용한 결합데이터의 정확도가 전체데이터의 성능보다 더 좋은 몇 가지 경우를 발견하였다. 이것은 본 논문에서 제안하는 방법을 고안할 때 생각하지 못한 특별한 경우이다. 이 성능들은 룰이 잘 만들어졌음을 시사하고 있는 것이다. 이 특별한 경우들은 우리가 시험을 통해서 얻은 성능 중에서 가장 좋은 성능을 보이고 있다.

앞으로, 이 방안을 다른 분류기에 적용하고 비슷한 결과를 얻는다면 이 방안이 다른 분류기에도 적용 가능함을 알 수 있을 것이다. 그리고, 본 논문에서 발견한 결합데이터의 정확도가 전체데이터의 성능보다 높은 경우를 발견하지 못한다면 이것은 또한 본 논문에서 사용한 분류기 유추(UChoo)의 유능함을 증명하는 것이 될 수 있다고 사료된다.

REFERENCES

[1] Mehmet Sabih Aksoy, “Pruning Decision Trees Using RULES3 Inductive Learning Algorithm”, *Mathematical and Computational Applications,* Vol. 10, No. 1, pp. 113-120, 2005.

[2] J.B. Larson, R.S. Michalski, “Selection of Most representative Training Examples and Incremental Generation of VL_1 Hypothesis: The Underlying Methodology and the Description of Programs ESEL and AQ11”, *Technical*

Report 867, Department of Computer Science, University of Illinois, May 1978.

[3] J.R. Quinlan, "Learning Efficient Classification Procedures and Their Application to Chess End Games", *Machine Learning*, Palo Alto: Tioga Press, 1983.

[4] D. H. Kim, D. H. Lee, and W. D. Lee, "Classifier using extended data expression," *IEEE Mountain Workshop on Adaptive and Learning Systems*, Logan:UT, pp. 154-159, 2006.

[5] D. H. Kim, D. H. Seo, and W. D. Lee, "Classifier Capable of Rule Refinement", *International Symposium on Computer Science and its Application*, Hobart, Australia, pp. 216-221, 2008.

[6] J. M. Kong, D. H. Seo and W. D. Lee, "Rule refinement with extended data expression," *IEEE Computer Society, Proceedings of the Sixth International Conference on Machine Learning and Applications (ICMLA)*, pp. 310-315, 2007.

[7] D. Oursten, R.J. Mooney, "Changing Rules: A Comprehensive Approach to Theory Refinement", Proceedings of the Eighth National Conference on Artificial Intelligence, Boston, MA, p.815, 1990.

[8] H. S. Jeon and W. D. Lee, "Pruning Method With Majority and Minority Properties," *International Conference on Information Science & Application (ICISA)*, in press, 2014.

[9] J. R. Quinlan, "C4.5: Program for Machine Learning", San Mateo, Calif, Morgan Kaufmann, 1993.

[10] J. R. Quinlan, "Bagging, Boosting, and C4.5", AAAI/IAAI, vol. 1, 1996.

[11] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, "Introduction to DATA MINING", Addison Wesley, pp. 207-312, 2005.

[12] UCI Repository of Machine Learning Databases [Internet]. Available: http://www.ics.uci.edu/~ml.

전해숙(Hae Sook Jeon)

1992: 충남대학교 전산학과 학사
1995: 충남대학교 컴퓨터과학과 석사
2008: 충남대학교 컴퓨터공학과 박사 수료
2000~현재 : 한국전자통신연구원 선임연구원
※관심분야 : 머신 러닝, 정보 과학, 네트워크

이원돈(Won Don Lee)

1979: 서울대학교 화학과 학사
1982: UIUC 화학과, 전산학과 석사
1986: UIUC 전산학과 박사
1986: UT Alington 교수
1987~현재 : 충남대학교 교수
※관심분야 : 머신 러닝, 멀티미디어, 금융공학