□기술해설□

ID3 계열의 귀납적 기계학습

숭실대학교 박영택* · 이강로**

목 차

1. 머리말

기계학습(machine learning)은 컴퓨터 프로그램을 보다 적응성(adaptive)있게 해주고 인공지능 프로그램이 극복하여야 할 지식습득 문제를 해결해 줄 가능성이 있기 때문에 인공지능이 시작한 시점부터 여러 가지 방식에 대한 연구가 꾸준히 진행되어 오고 있다 [2][17][18][28][30][32]. Simon은 인공지능에서의 학습을 정의하면서, 프로그램이 하나의 타스크를 수행한 후에 그 추론 과정에서 얻은 경험을 바탕으로 시스템의 지식을 수정 및 보완하여 다음에 그 타스크나 또는 비슷한 타스크를 수행할 때는 처음보다는 더 효율적이고 효과적으로 문제를 해결할 수 있는 적응성을 학습기능이라고 지칭하였다 [29].

인공지능 프로그램이 보유하여야 할 대표적인 기능 중의 하나는 주어진 상황에 적절하게 적응하면서 같은 실수를 반복하지 말아야 한다는 점이다. 이와 같은 목적을 위해서는 다양한 기계학습 프로그램이 추론기관과 상호 보완적으로 작용할 필요가 있다. 그림 1이 보여주는 바와 같이 추론기관은 지식베이스를 이용하여 문제를 해결하는 반면에 기계학습 시스템은 지식베이스에 대한 여러 가지 작업을 수행하게 된다. 즉, 기계학습 시스템이 수행할 수 있는 첫 번째의 기능은 외부로부터 필요한 지식을 추출하여 초기 지식베이스를 구축하는 기능을 들 수 있다. 일반적으로 인공지능 프로그램에 필요한 지식베이스는 지식공학자가 전문가를 인터뷰하는 과정을 통해서 구축된다. 그러나, 이와 같은 수작업 과정의 많은 부분은 기계학습에 의해서 해결이 될 수 있다. 예를 들면, 의사나 컴퓨터 고장진단 전문가가 환자 또는 컴퓨터를 진단한 과정을 사례로 구축할 수 있는 데 이와 같은 수많은 진단 사례로부터 추론기관이 활용할 수 있는 진단 지식을 추출하는 과정에 기계학습이 이용될 수 있다.

지식베이스는 특성상 항상 불완전하고(incomplete), 모순성이 있으며(inconsistent) 또는

*종신회원
**학생회원

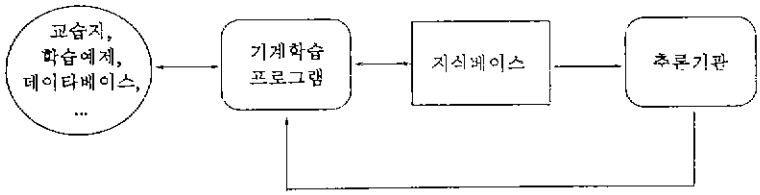


그림 1 인공지능 프로그램에서의 기계학습

부정확(incorrect)한 경우가 생기게 되고 이와 같은 지식베이스를 활용하는 추론기관은 주어진 문제를 제대로 해결하지 못하는 경우가 생긴다. 이러한 추론기관의 실패(failure)를 분석하고 지식베이스를 수정하고 보완하는 과정은 시스템 개발자가 지식베이스를 디버깅하는 과정에 해당하는 데 기계학습은 이와 같은 문제점을 극복하기 위한 방법을 제공하기도 한다 [9].

기계학습의 활용은 다음과 같은 점에서 시스템을 보다 지능화 또는 적응성이 있도록 한다. 첫번째는, 대부분의 인공지능 시스템 또는 지능형 시스템들은 지식베이스를 이용하고 있고 지식베이스를 구축하는 데는 지식습득 병목현상에 의해서 시스템 구축의 한계가 있다. 이러한 문제를 극복하는 도구로 기계학습은 많은 부분을 반자동으로 하는 데 기여할 수 있다.

두번째로는, 대부분의 지능형 시스템들은 사용자와 상호 작용을 하면서 사용자의 행위에 대한 특성이나 사용자의 취향을 알아내어 가장 적절한 정보를 공급하거나 또는 가장 적절한 인터페이스를 제공하는 것을 목적으로 한다. 이와 같은 목적을 가진 시스템들이라면 사용자의 의도를 정확하게 파악하는 수단 중의 하나로 기계학습을 이용하는 것이 필연적일 것이다. 기계학습은 사용자의 행위를 개념적으로 특성화하는 과정을 통해서 사용자의 의도를 파악하는 기능을 수행하고, 따라서 시스템을 보다 사용자의 특성에 적응성있게 구축할 수 있게 한다.

세번째로는, 대부분의 지능형 시스템이 보유하여야 할 특성 중의 하나는 과거의 실수를 되풀이하지 않도록 지식베이스를 일관성 있게 수정할 수 있는 능력이 있어야 한다. 또한, 지식베이스를 수정한 과정에 대한 설명을 제공할 수 있을 때, 일관성 있게 지식베이스를 보완할 수 있으므로 인공지능 기법의 기계학습을 이용한 시스템의 구축이 필요하다. 이와 비슷하게, 지능형 시스템이라면 처음에 문제를 해결하는 과정보다 두번째에 같은 문제를 해결할 때는 보다 효율적이어야 한다. 이러한 효율성을 위해서는 지식베이스를 컴파일 하여 보다 효과적으로 문제를 해결할 수 있는 기능을 가질 필요가 있는데 이러한 기능을 기계학습이 제공할 수 있다.

2. 기계학습의 종류

기계학습의 종류는 보는 사람의 관점에 따라서 다를 수가 있으나, 한가지 방식을 사용하는 단일전략(mono strategy) 방식과 다전략(multi strategy) 방식으로 구별될 수 있다. 기계학습에 대한 연구는 인공지능의 시작과 함께 진행되어 오면서 1980년대 중반까지는 귀납적(inductive) 방식에 대한 연구가 활발히 수행되어 왔었고, 1980년대 중반 이후 1990년대 초반까지는 연역적(deductive) 방식에 대한 연구가 다양한 각도로 이루어졌다.

귀납적 학습은 예제들 간의 유사성(similarity)을 기반으로 예제가 암시적으로 나타내는 개념을 추출하는 방법을 사용하므로 흔히 유사성기반의 학습(SBL : Similarity-based learning)이라고도 불린다 [19][20]. 반면에, 연역적 방식은 domain 이론을 이용하여 하나의 예제로부터 보다 효율적인 지식을 생성한다. 이러한 특성으로 인해서 domain 이론이 완전하고(complete), 일관성이 있으며(consistent), 정확하다면(correct) 연역적 방식은 주어진 예제를 이용하여 domain 지식을 보다 활용성이 있는 지식으로 변환시켜 준다. 이러한 과정은 proof 트리를 구축하면서 주어진 예제가 어떻게 개념(concept)에 만족될 수 있는 가를 설명하기 때문에 설명기반 학습(EBL : Explanation-based

learning)이라는 이름으로 연구가 진행되어 왔다 [6][23].

다전략 방식은 귀납적 및 연역적 기계학습을 혼용하므로써 원하는 목적을 이루는 방식으로 각각의 기계학습에 대한 연구가 충분히 진행된 1990년대부터 많이 연구되어 왔다. 학습이라는 과정이 어느 한 방식만으로 이루어지기는 힘들므로 필요에 따라서 다양한 기계학습을 유효적절하게 사용하는 다전략 방식이 경우에 따라서는 효율적일 수 있다 [12][28].

또한, 기계학습은 외부에서 학습 예제를 제공하는 과정에서 각각의 예제에 대한 클래스 및 기타 부수 정보를 제공하는 대상이 있는 경우를 supervised 방식이라고 하고 그렇지 않은 경우를 unsupervised 방식이라고 구별한다. 그러므로, 귀납적 방식도 학습 예제를 제공하는 환경에 따라서 supervised 귀납적 기계학습과 unsupervised 귀납적 방식으로 구별된다 [8]. 본 장에서는 이와 같은 기계학습을 귀납적, 연역적 및 사례기반으로 구별하여 개략적으로 설명하고자 한다.

2.1 귀납적 기계학습

귀납적 기계학습은 주어진 예제들의 유사성을 이용하여 이들이 암시적으로 표현하고 있는 가설을 추출한다. 귀납적 학습은 얻고자 하는 개념이 포함하여야 할 양의 예제(positive examples)와 포함하지 말아야 할 음의 예제(negative examples)를 입력하여 이를 처리한다. 즉, 귀납적 방식은 모든 양의 예제는 포함하지만 어떤 음의 예제도 포함하지 않는 개념을 얻기 위해서 일반화(generalization) 과정을 수행하여 가설을 생성하고 이를 기반으로 새로운 예제가 입력되었을 때 그 예제의 클래스를 예측할 수 있게 된다. 이와 같은 학습과정의 예는 3장과 4장에서 구체적인 예를 가지고 설명하겠다.

일반적으로 귀납적 학습은 주어진 입력 예제를 보고 음의 예제는 포함하지 않으면서 양의 예제를 포함하는 특성을 알아내는 작업을 수행한다. 예를 들면 그림 2에 있는 바와 같이 다양한 컵의 예제들이 입력되었다면 귀납적 방식은 컵을 표현하는 속성들 간의 상호 유사성을 고려하여 컵을 표현하기 위한 속성과 속성값을 정하게 된다. 그림 2에서 보여지는 것과 같이 컵을 표현하는 데 불필요한 속성인 color와 같은 것은 귀납적 학습이 생성한 컵의 정의에 포함되지 않는다.

유사성 기반의 귀납적 학습은 명시적인 domain 이론을 이용하지 않고 다분히 경험적인 방식으로 개념을 표현하는 description을 추출한다. 그러나, 대부분의 귀납적 학습에서는 특정한 개념을 원활하게 구하기 위해서 나름대로의 암시적인 bias를 이용하고 있다 [11][31]. 일반적으로 귀납적 기계학습에서는 bias 입력 예제와 학습되어질 개념을 표현하는 언어, 각각의 속성들이 가지는 값들에 대한 계층적 구조와 같은 내용이 된다.

feature	CUP-1	CUP-2	NON_CUP-1
color	blue	red	blue
owner	kim	park	choi
made-of	ceramic	ceramic	ceramic
flat-bottom?	yes	yes	yes
concavity?	yes	yes	yes
handle?	yes	yes	yes

(a) SBL의 입력 예

CUP

made-of	ceramic
flat-bottom?	yes
concavity?	yes
handle?	yes

(b) SBL에 의해서 생성된 컵의 일반적 정의

그림 2 유사성 기반 학습의 예

2.2 연역적 기계학습

연역적 기계학습은 귀납적 방식과는 다르게 domain 이론을 이용하여 학습을 수행한다. 귀납적 방식이 많은 예제들과 귀납적 bias을 이용하는 반면에, 연역적 기계학습은 domain이론을 이용하고 하나의 예제만을 필요로 한다는 점이 다르다. 즉, 귀납적 방식은 경험적인 방식으로 가설을 생성하는 과정을 취하고 있지만, 연역적 방식은 domain 이론에 근거하여 하나의 주어진 예제를 설명하면서 개념적인 domain 지식을 보다 구체적이고 활용성이 있는 지식으로 컴파일한다.

귀납적 기계학습은 새로운 가설을 생성하면서 지식베이스에 지식을 추가하는 반면에, 연역적 기계학습은 주어진 지식베이스를 보다 효율적인 형태로 변환하는 특성이 있다. 그러므로, 지식의 범위라는 측면에서 보면 연역적 방식이 새로운 지식을 추가하지는 못하더라도 기존의 지식을 보다 쉽게 적용할 수 있는 형태로 바꾸므로써 추론의 속도를 개선할 수 있게 된다 [9]. 학습의 정의가 같은 타스크를 처음 수행할 때보다 두번째 이후에 실행할 때는 보다 빠르게 할 수 있는 기능을 포함하고 실제로 많은 경우의 학습이 이와 같은 특성을 가지고 있으므로 연역적 기계학습은 중요한 의미를 가진다. 일반적으로 연역적으로 기계학습은 proof 트리를 이용하여 예제를 설명하고 이 설명을 일반화하므로 설명기반 학습(EBL, EBG : Explanation-based learning, Explanation-based generalization) 이라고 한다 [3][6][21][23]. 또한, EBL은 많은 macro와 chunk [15][16]를 생성하고 이를 이용하므로써 추론의 속도를 개선하므로 speed-up 학습이라고 불리지만 불필요한 macro와 chunk로 인한 utility문제도 심각한 장애요소가 된다.

보다 구체적인 연역적 기계학습에 대해서는 본 정보과학회지에 실린 두편에 EBL에 관한 논문을 참조하기 바란다.

2.3 사례기반 기계학습

사례기반 학습(Case-based learning)은 일정한 분야에 적용할 수 있는 대표적인 사례들을 기억하고 있다가 새로운 문제를 해결하는 과정에서 유사한 사례를 이용하여 문제를 해결한다. 이 과정에서 주어진 사례들을 가지고 당면한 문제를 제대로 해결하지 못한 경우는 그 문제를 새로운 사례로 기억하였다가 추후에 이용한다는 점에서 학습의 기능을 가진다. 이와 같은 사례기반 학습은 일단 주어진 문제가 어떤 사례와 유사한가를 알아내는 indexing 및 유사성 비교와 같은 과정을 필요로 한다. 또한, 새로운 사례를 기억하고 학습하기 위해서는 당면한 문제를 사례로 기억할 것인가 또는 무시할 것인가 하는 결정, 어떤 indexing 지식이 기억될 사례에 필요한가를 결정하는 문제, 추후에 사례와 다른 문제간의 유사도를 계산하기 위해서는 어떤 domain 지식이 필요한가를 결정하는 문제, 학습된 사례를 일반화하는 문제와 같은 부수적인 과정을 필요로 한다 [10][13][14]. 보다 구체적인 사례기반 학습에 대해서는 본 정보과학회지에 실린 사례기반에 대한 논문을 참조하기 바란다.

3. ID3 계열의 귀납적 기계학습

귀납적 기계학습에는 여러 종류가 있으나 본 논문에서는 ID3 계열의 기계학습에 대해서 살펴보고자 한다. ID3 계열의 기계학습 시스템들은 많은 예제들로부터 그 예제들이 암시적으로 포함하고 있는 개념을 추출하여 결정트리(decision tree)의 형태로 일반화하고 이를 이용하고 새로운 예제들을 분류하는 기능을 가지고 있다 [24][25][26][27]. 결정트리는 노드와 링크로 구성된 일반적인 트리로서, 각각의 노드는 예제들이 표현된 속성(attribute)중의 어느 하나가 된다. 또한, 링크는 그 속성을 테스트하였을 속성이 취할 수 있는 값을 의미하게 된다. 그림3은 간단한 결정트리를 보여주고 있다. 그림에서 알 수 있는 바와 같이, 모든 예제는 그 예제가 속하는 클래스가 결정되고, 각각의 노드는 속성을 의미하고 속성값에 따라서 서로 다른 링크로 분리되므로, 트리의 root로부터 터미널 노드까지의 path는 하나의 규칙으로 변환이 가능하다. 예를 들면, 그림3에 있는 트리를 depth-first-order로 traverse한 경우에 가장 먼저 만나는 leaf는 다음과 같은 규칙으로 변환이 가능

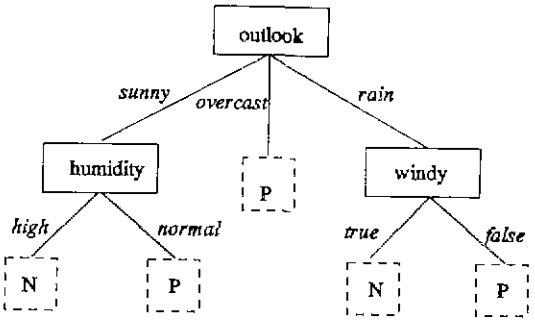


그림 3 결정트리의 예

하다.

if (outlook = sunny) & (humidity = high) then N.

ID3 계열의 귀납적 학습프로그램들은, 많은 예제들로부터 이와 같은 결정트리를 만드는 것을 목적으로 한다. 결정트리는 주어진 예제들을 올바르게 분류할 뿐만이 아니라, 주어진 예제의 수가 충분히 많은 경우는 새로운 예제가 들어오는 경우에 과거의 예제들이 가지고 있던 정보에 가장 근사한 클래스로 분류할 수 있는 학습 능력을 가지게 된다. 많은 예제들이 주어졌을 때, 그들을 올바르게 분류할 수 있는 결정트리는 여러 가지가 있을 수 있다.

그러므로, ID3는 가장 간단한 트리를 구성하는 것을 목적으로 하고 있다. 여기서 간단한 결정트리는 적은 수의 테스트만으로도 주어진 예제들을 올바르게 분류할 수 있는 트리를 의미한다.

3.1 ID3의 학습 예

ID3학습을 위한 입력은 클래스에 따른 양의 예제(positive example)와 음의 예제(negative example)로 구별이 된다. 또한, 각각의 예제는 속성(attribute)과 속성값으로 이루어진 쌍(pair)의 집합으로 표현되어진다. 이와 같은 속성은 일정한 수로 미리 정의되어진다.

그러므로, ID3를 이용하여 학습을 하는 경우는 최소이며 가장 효율적으로 예제를 표현하기에 적절한 속성을 정의하는 작업이 필요하다. ID3는 예제 집합을 입력하여 터미널 노드에 한 클래스만이 존재할 때까지 결정 트리를 구성하게 된다.

이와 같은 ID3의 프로세스를 설명하기 위해서 다음과 같은 예제들과 이를 이용한 ID3의 처리과정를 살펴보기로 한다. 설명을 위해서 클래스는 +, − 두개로 구분되는 것을 가정한다. 즉, 모든 예제들은 + 또는 − 클래스로 분류되어 입력된다. 입력 예제는 3개의 속성인 height, hair, eyes로 표현되고 각각의 속성이 가지는 속성 값은 다음과 같다고 가정을 하자.

height = [short, tall]
hair = [dark, red, brown]
eyes = [blue, brown]

또한, 다음과 같은 8개의 예제들이 입력되었다고 가정을 하자. 즉, 첫번째의 예제는 (height = short, hair = blond, eyes = blue) 이면 + 클래스라는 것을 의미한다.

height	hair	eyes	
short	blond	blue	+
tall	blond	brown	-
tall	red	blue	+
short	dark	blue	-
tall	dark	blue	-
tall	blond	blue	+
tall	dark	brown	-
short	blond	brown	-

그림 4 ID3의 입력 예제

이와 같은 예제 입력들을 이용하여 여러 종류의 결정트리를 구성할 수 있으나, ID3는 가장 간단한 결정트리를 그림 5와 같이 구성한다. 앞에서도 설명한 바와 같이 터미널 노드는 + 또는 − 클래스에 속하는 예제들로만 분류되어진다. 이러한 결정트리에서 얻을 수 있는 귀납적인 가설은 4가지로 구성된다. 즉, hair가 dark이면 − 클래스, hair가 red이면 + 클래스, hair가 blond이고 eyes가 blue이면 + 클래스, hair가 blond이고 eyes가 brown이면 − 클래스라는 네개의 가설이 생성된다. 이러한 결정트리에 의해서 + 클래스를 위한 개념은 다음과 같은 식으로 표현이 될 수 있다.

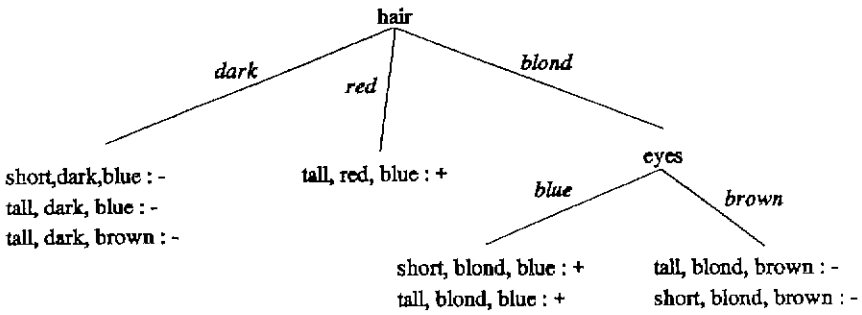


그림 5 ID3에 의해서 구성된 결정트리

(hair = red) ∪ [(hair = blond) & (eyes = blue)]

ID3는 귀납적인 방식을 취하고 있으므로 이와 같은 가설은 항상 틀릴 수가 있지만 충분한 예제를 가지고 결정트리를 구성하는 경우는 새로운 예제를 거의 올바르게 분류할 수 있게 된다. 예를 들면, 그림4에 있지 않은 새로운 예제인 (height = short, hair = red, eyes = brown)가 들어오는 경우에 위의 결정트리를 이용하면 첫번째의 속성인 hair에 의해서 + 클래스로 분류가 된다. 이러한 특성은 충분한 예제로부터 학습된 결정트리는 새로운 예제를 분류하는데 이용될 수 있다는 것을 의미한다.

3.2 ID3 알고리즘

ID3는 주어진 예제들로부터 양의 예제는 모두 포함하면서 음의 예제는 하나도 포함하지 않는 결정트리 중에서 가장 간단한 형태의 결정트리를 생성하는 것을 목적으로 한다. 이와 같은 결정트리를 구한다는 것은 NP 문제에 속하게 되므로 ID3은 정보이론 (information theory)에 따른 엔트로피 개념을 이용하여 가장 간단한 결정트리를 구하고 있다. 일반적으로 엔트로피는 무질서도를 나타내는 정량적인 수치이다. 일반적으로 ID3에 입력된 입력 예제 집합은 양의 예제와 음의 예제가 혼합되어 있으므로 어떤 속성이 어떻게 예제를 분류하는 지에 대한 정보가 없기 때문에 엔트로피가 아주 높은 상태이다. 그러나, 앞의 예제에서 볼 수 있듯이 일단 결정트리가 학습된 상태에서는, 각각의 터미널 노드가 하나의 클래스로만 결정이 되므로 무질서도는 0이 된다.

그러므로, ID3가 예제를 모두 분류한 상태에서는 엔트로피가 0이 되는 상태를 의미하므로 ID3가 예제를 모드 분류한 상태에서는 엔트로피를 최소화하는 과정을 반복한다.

ID3에서 이용하고 있는 무질서의 값은 다음과 같다.

무질서도 $I(p, n) =$

$$-\frac{p}{p+n}\log\frac{p}{p+n}-\frac{p}{p+n}\log\frac{p}{p+n}\log \quad (1)$$

식 1에서 p, n은 각각 양의 예제의 갯수와 음의 예제의 갯수를 의미한다. 그러므로 $(p/(p+n))$, $(n/(p+n))$은 각각 양의 예제 및 음의 예제일 확률을 의미한다.

이해를 돕기 위해서 그림6과 같은 경우를 보자. 첫번째 단계에서는 양의 예제가 2개, 음의 예제가 2개가 있으므로 homogeneous하지 못한 특성을 가지고 있다. 이와 같은 점을 식1를 이용하여 정량화하면 (-0.5 log 0.5 -0.5 log 0.5)가 된다. 만약에 이와 같은 예제들을 분류하는데 eyes라는 속성을 이용하였다면 그림6에서 보여주는 바와 같이 eyes의 값에 따라 두개의 그룹으로 나뉘어지면서 터미널 노드는 homogeneous한 두개의 그룹으로 나뉘어 진다. 각각의 그룹의 무질서도를 위의 식으로 정량화하면 각각 0이 되어서 무질서도가 감소하게 된다.

이와같은 ID3의 학습 알고리즘은 다음과 같은 프로시듀어로 설명된다.

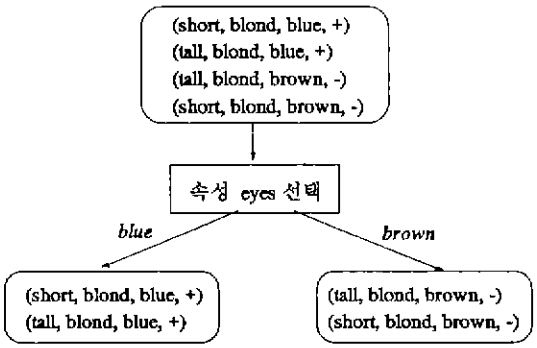


그림 6 속성의 선택으로 인한 무질서도의 감소

1. 입력 예제의 집합을 C라고 가정한다.
2. C가 양의 예제들로만 구성되어 있으면 + 노드를 만들고 끝낸다.
3. C가 음의 예제들로만 구성되어 있으면 − 노드를 만들고 끝낸다.
4. C가 heterogenous한 예제들로 구성되어 있으면 다음과 같은 과정을 수행한다.

4.1. 속성 중에서 선택되었을때에 무질서도를 최대한으로 줄일 수 있는 속성 F를 선정한다.

4.2 속성 F가 가지는 값에 따라서 C를 여러 개의 subset인 $C_1, C_2, \ldots C_n$으로 분류한다.

4.3 각각의 C_i에 재귀적으로 ID3 알고리즘을 적용한다.

위의 프로시듀어 중에서 4.1에서 수행하는 속성을 선택하는 과정이 ID3의 주요한 기능 중의 하나라고 할 수 있다. ID3에서는 입력된 예제들을 최소의 테스트로 분류할 수 있는 결정트리를 구성하기 위해서 information-theoretic한 방식을 취하고 있다. 즉, 현재 상태의 무질서도는 일정하게 주어지므로, 속성을 선택함에 따라서 여러 종류로 분류가 가능하고 그에 따라서 무질서도가 감소하게 된다. ID3는 이 중에서 무질서도를 가장 많이 감소할 수 있는 속성을 선택하여 결정트리를 구성하게 된다. 예를 들면 그림 7에서 현재의 상태가 C이고 C의 무질서도가 I(p, n) 이라고 가정을 하자. 여기서 분류에 이용되는 속성이 f_1, f_2, f_3가 있다고 가정을 할 때 각각의 속성에 따라서 분류를 해보고 분류 결과의 무질서도가 가장 감소한 속성이 1의 F로 선정이 된다.

그림 7에서 속성을 선택하고 난 후의 각각의 C_i에 대한 무질서도를 취하고 이에 가중치를 곱하면 다음과 같은 무질서도의 정략적인 값을 구할 수 있게 된다.

$$E(A) = \sum_{i=1}^{n} \frac{pi + ni}{p + n} I(pi, ni) \qquad (2)$$

식2에서 I(pi, ni)는 C_i에 대한 무질서도를 구하는 식으로 C_i에 있는 양의 예제 및 음의 예제 수를 이용한 확률식으로 구할 수 있다. 또한, (pi + ni)/(p + n)은 각각의 C_i에 대한 가중치를 의미한다. 모집합 C는 양의 예제 p개, 음의 예제 n개를 가지고 있었으므로 임의의 속성에 의해서 생성되는 C_i들의 모든 예제의 수는 항상 p + n개가 된다. 또한, C_i그룹의 양의 예제가 pi개, 음의 예제가 ni개가 있다고 가정을 하면 (pi + ni)/(p + n)는 C_i가 가지는 가중치가 된다. 그러므로, ID3는 C의 무질서도 I(p,n)과 E(A)과의 차이를 최대로 하는 속성을 선택하여 결정트리를 구하게 된다. 이와 같은 과정은 무질서도를 급격히 감소시킬 수 있는 속성을

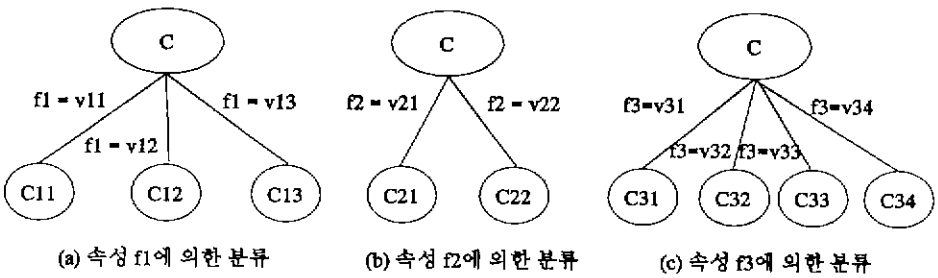


그림 7 가능한 속성에 의한 분류

선택하므로써 가장 적은 수의 테스트만으로 예제들을 분류할 수 있는 결정트리를 구하게 된다.

만약의 그림7에서 f_1을 이용하는 경우에 식 2를 이용하여 C_{11}, C_{12}, C_{13}의 엔트로피를 구한 값이 E_1, f_2를 이용한 경우는 E_2, f_3를 이용한 경우가 E_3라고 가정을 하자. 이때 $E_1 < E_2 < E_3$ 라는 관계가 존재한다면 f_1을 선택하였을 때 무질서도를 가장 최소로 만들 수 있으므로 이 단계에서는 f_1를 이용하여 결정트리를 구하는 것이 바람직하다. ID3는 이와 같은 과정을 재귀적으로 C_{11}, C_{12}, C_{13}에 적용하게 된다.

4. ID5 귀납적 기계학습

귀납적 기계학습은 점진적 학습과 비점진적 학습으로 나눌 수 있는데, 이점에서 ID3와 ID5의 차이점을 볼 수 있다. ID3는 비점진적인 학습 방법을 사용하므로써 새로운 예제에 대한 학습을 위해서는 이전의 모든 학습예제를 다시 참조하여야 한다. 즉, 현재 학습되어 있는 상태에 새로운 예제가 입력되었을 경우 과거에 사용했던 예제에 새로 입력된 예제를 추가하여 다시 학습하는 방식을 사용한다. ID5에서는 현재의 결정트리를 새로운 예제가 입력되었을 경우 필요하다면 교정하는 형식을 취한다 [24][25][26]. 즉, ID3와는 달리 점진적인 학습을 할 수 있기 때문에 새로운 학습예제가 입력되었을 경우 이 학습 예제가 현재 구성되어 있는 결정트리에 의해 분류될 수 없는 경우에 한해서 현재의 결정트리를 교정한다. 따라서, 비점진적 학습방식은 동적으로 계속 추가될 수 있는 일련의 학습 예제들이 있을 경우 적합한 학습 방식이다. 그러므로, 계속 새로운 학습 예제가 입력되는 경우 과거의 학습 예제를 다시 참조할 필요 없이 현재 구성되어 있는 결정트리를 교정하는 것으로 새로운 결정트리를 생성할 수 있는 점진적인 학습 방식이 좀 더 효율적이다. 즉, 일련의 계속적인 예제들을 학습하는데에 점진적인 학습 방식을 사용하므로써 좀 더 효율적인 기계학습을 할 수 있다.

4.1 ID5의 학습 예

학습에 사용된 예제는 그림8에 있는 바와 같이 [+, −] 2개의 클래스와 [height, hair, eyes] 3개의 속성을 갖는다. 본 절에서는 그림8의 예제가 하나씩 입력된다고 가정을 한다. 3장의 ID3는 모든 예제가 한번에 입력되어 처리하는 비점진적 방식이었지만 ID5는 예제를 하나씩 처리하는 점진적 방식으로 본 절에서는 이와 같은 특성을 보여주고자 한다.

Class	height	hair	eyes
-	short	blond	brown
-	tall	dark	brown
+	tall	blond	blue
-	tall	dark	blue
-	short	dark	blue
+	tall	red	blue
-	tall	blond	brown
+	short	blond	blue

그림 8 ID5의 입력 예제

첫번째 예제 (−, height=short, hair=blond, eyes=brown)이 입력되었을 경우 결정트리는 하나의 터미날노드를 갖게 된다.

(eyes=brown, hair=blond, height=short)

그림 9 첫번째 예제 입력

두번째 예제 (−, height=tall, hair=dark, eyes=brown) 역시 클래스가 − 이기 때문에 터미날 노드에 삽입된다.

(eyes=brown, hair=dark, height=tall)
(eyes=brown, hair=blond, height=short)

그림 10 두번째 예제 입력

세번째 예제는 클래스가 + 이기 때문에 결정트리가 확장된다. 이때 결정트리를 확장하는

대 가장 좋은 속성값을 선택하여 트리를 확장하게 된다.

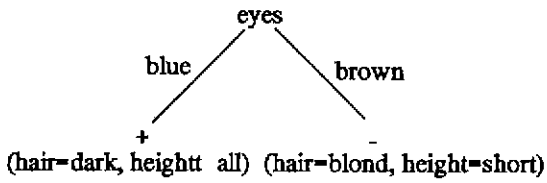


그림 11 세번째 예제 입력

네번째 예제 (−, height=tall, hair=dark, eyes=blue)가 입력되었을 경우 결정트리가 클래스를 구별할 수 없기 때문에 현재 구성되어 있는 결정트리를 확장하게 된다.

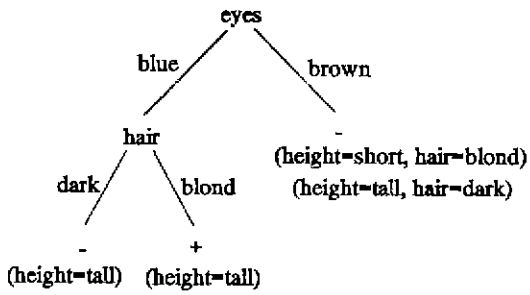


그림 12 네번째 예제 입력

다섯번째 예제 (−, height=short, hair=dark, eyes=blue)가 입력되었을 경우 현재의 결정트리로 클래스를 구별할 수 있으나 결정트리를 구성하는데 노드에서 가장 좋은 속성값이 eyes가 아닌 hair로 바뀌게 된다. 따라서 결정트리에서 속성들의 위치를 서로 바꾸는 작업을 수행하게 된다. 즉, 루트 노드의 eyes대신 hair가 가장 좋은 속성값으로 자리를 잡게 된다. 이렇게 결정트리가 다시 재구성 되는 과정은 다음에서 살펴보기로 한다.

위에서 보여진 바와 같은 과정을 거쳐 8개의 예제를 학습한 결과는 그림13과 같다.

4.2 결정트리 재구성 예

결정트리를 확장해 가는 과정에서 가장 좋은 속성값이 다를 경우 결정트리를 재구성 하게 되는데 그림12에 (−, height=short, hair=dark, eyes=blue)의 예제가 입력되면 그림14와 같은 결정트리가 된다. 이때 eyes보다 hair가 결정트리를 구성하기에 더 좋은 속성값이 된다. 이 경우 결정트리를 재구성하게 되는데 그 과정은 다음과 같다.

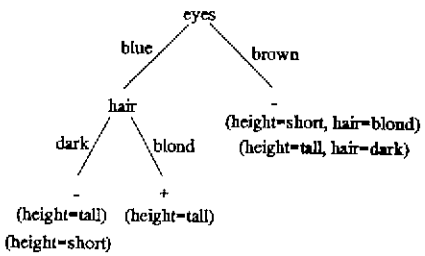


그림 14 재구성 되기 이전의 결정트리

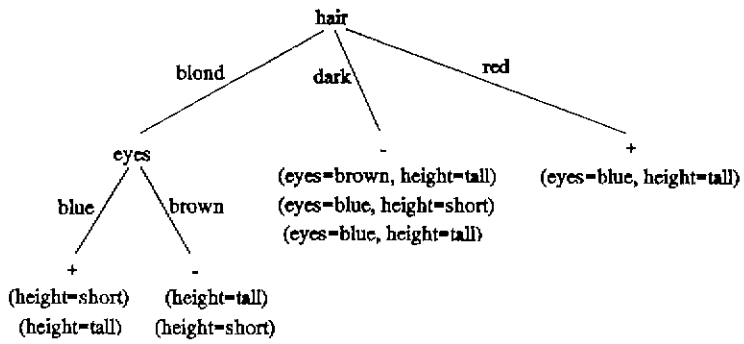


그림 13 8개의 예제를 학습한 결정트리

1. eyes에 대해 결정트리를 분할 한다.

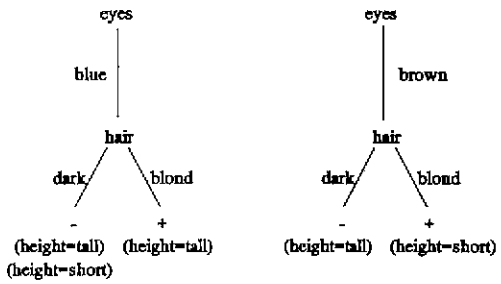


그림 15 두개의 분리된 결정트리

2. hair와 eyes를 서로 바꾼다.

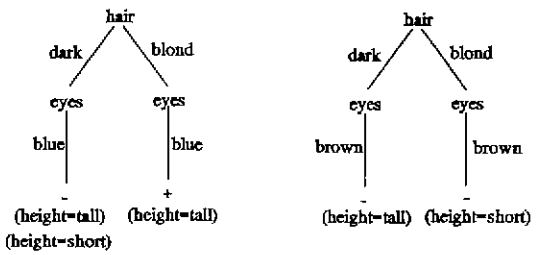


그림 16 두개의 속성이 서로 바뀐 상태의 결정트리

3. 분할된 결정트리를 다시 합한다.

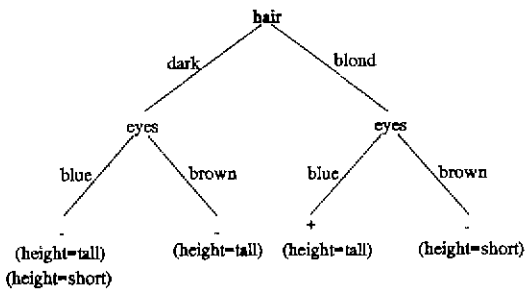


그림 17 분할된 결정트리가 합쳐진 상태

4. 자식 노드가 모두 같은 클래스일 경우 하나로 합한다.

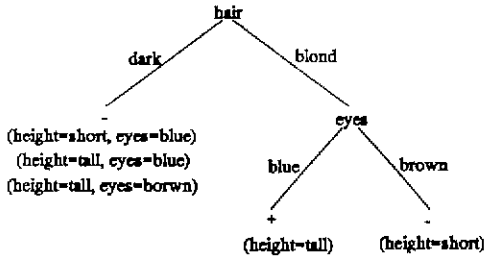


그림 18 재구성이 완료된 결정트리

4.3 ID5 알고리즘

점진적으로 결정트리를 구성하기 위한 ID5 알고리즘은 다음과 같다.

1. 각 노드에서 속성에 대해 클래스의 카운트 값을 변경시키고, 필요에 따라 결정트리를 확장한다.
2. 만약 현재 노드의 속성이 가장 좋은 속성(A_{best})이 아닐 경우 현재 노드의 속성과 가장 좋은 속성을 서로 바꾼다.
3. 만약 현재 노드가 하나의 클래스만을 구성하는 터미날 노드일 경우 학습 예제의 나머지 부분을 저장하고 끝낸다.
4. 재귀적으로 A_{best} 하위 결정트리 부분을 수정해 나간다.

결정트리를 구성하는데 노드의 속성을 서로 바꾸어 결정트리를 재구성해야 할 경우의 알고리즘은 다음과 같다.

1. 재귀적으로 속성을 바로 아래 서브트리의 루트로 끌어 올린다. (필요에 따라서는 서브트리를 확장한다)
2. 각 서브트리에 대해 새로운 속성과 이전의 속성을 서로 바꾸어 새로운 결정트리를 생성한다.
3. 서브트리를 합한다.

위에서 언급한 결정트리 구성을 위한 알고리즘과 결정트리의 재구성을 해야할 경우 필요한 알고리즘은 앞에서 그림으로 보여진 결정트리의 구성 과정과 재구성과정을 보면 쉽게 이해할 수 있을 것이다.

5. 귀납적 기계학습의 응용

귀납적 기계학습은 여러 가지 형태로 실제

응용 시스템에 적용되고 있다. 특히 본 논문에서 설명한 ID3 계열의 프로그램들은 공용 소프트웨어로서 누구나 용도에 따라서 활용이 가능하다. 귀납적 기계학습이 유용하게 쓰이고 있는 분야는 여러 가지가 있으나 본 절에서는 크게 초기지식베이스 구축과 지능형 에이전트에서의 귀납적 기계학습의 활용도에 대해서 살펴보기로 한다.

5.1 지식습득 과정과 기계학습

지식베이스를 구축하는 과정은 크게 3단계로 구분된다. 첫번째 단계는 초기지식베이스를 구축하는 단계이고, 두번째 단계는 초기지식베이스가 완전할 수가 없으므로 주어진 테스트 케이스를 이용하여 지식베이스를 디버깅하는 단계이며, 세번째 단계는 주어진 지식베이스를 최적화하므로써 추출 속도를 개선하는 단계이다 [1].

첫번째 단계에서는 일반적으로 지식공학자가 전문가를 인터뷰하고 그 과정에서 얻는 지식을 지식베이스 설계자에게 전달하고, 설계자는 시스템의 용어를 이용하여 지식베이스를 구축하게 된다. 이와 같은 과정은 매우 힘들고 어렵기 때문에 일반적으로 지식습득 병목현상이 있게 되어서 인공지능 시스템 구축에 커다란 걸림돌이 되고 있다.

본 논문에서 설명한 귀납적 기계학습 방식들은 초기지식베이스를 구축하는 과정에 성공적으로 이용되어 왔다. 즉, 전문가를 인터뷰하는 대신에 현장에서 문제를 해결하는 과정을 기록하여 많은 예제나 사례를 만들 수 있다면 그것을 이용하여 초기지식베이스를 구축할 수 있다. 이와 같이 현장에서 기록을 하는 경우는 매우 흔히 있는 일이다. 예를 들면, 임상병리학교실과 같은 곳에서는 하루에도 수십건 이상의 새로운 환자의 기록이 만들어질 수 있으며, 원자력발전소나 일반적인 발전소에서 근무하는 운전원들은 매일 사고일지를 작성하게 되어 있고, 교환기나 통신망을 운용하는 운용자들로 교환기나 망의 fault, alarm 메세지와 그 처리 명령 및 반응 등에 대한 기록을 하고 있다.

이와 같이 수집된 많은 예제들은 귀납적 기계학습의 좋은 입력자료가 되고 수많은 입력자료를 이용하려 귀납적 기계학습시스템은 양질의 초기지식베이스를 손쉽게 구축할 수 있다. 본 논문에서도 보여주었듯이 ID3 계열의 프로그램들은 입력예제들을 설명할 수 있는 결정트리를 생성하게 되고 이와 같은 결정트리는 규칙을 만드는 데 이용될 수 있다. 이와 같은 귀납적 기계학습을 사용하는 경우에 가장 중요한 점은 입력데이타와 출력규칙을 표현할 용어를 정의하는 일이다. 일반적으로 용어를 정의하는 과정에서도 최소의 법칙을 적용하고 또한 용어가 결론은 추론기반이 이용할 규칙의 구성요소가 되므로 이를 고려하여 선정되어야 한다.

지식습득의 두번째 단계는 지식베이스를 디버깅하는 과정을 의미한다. 이러한 과정에서는 추론기관이 주어진 지식베이스를 이용하여 테스트케이스를 수행하여 그 결과의 옳고 그름에 따라서 지식베이스를 수정 또는 보완한다. 일반적으로 이와 같은 과정을 시스템 개발자가 추론기관과 지식베이스의 지식을 활용하여 지식베이스를 디버깅한다. 이러한 과정을 완전 자동화한다는 것은 불가능하지만 사례기반 방식이나 견습학습(apprenticeship learning)같은 방식을 도입하면 특정한 타스크에서는 추론기관의 실수(failure)를 극복하는 과정을 반자동화하는 것이 가능하다 [4][5][7][9][10].

세번째 단계는 지식베이스를 최적화하는 단계인데 이것은 지식베이스를 좀 더 효율적인 형태로 변환하는 것을 의미한다. 이러한 과정을 지식베이스에 있는 규칙들의 마크로나 chunk를 형성하고 이를 활용하면 가능한데 이것은 설명기반학습과 같은 연역적 방식이 자주 이용되고 있다.

5.2 지능형 에이전트와 귀납적 기계학습

지능형 에이전트는 사용자의 불충분한 요구사항을 인지하고 사용자의 의도에 맞는 goal을 알아내어 수행해 주는 소프트웨어를 지칭한다. 이와 같은 지능형 에이전트는 사용자의 행위를 보면서 취향을 파악하는 능력이 있어야 한다. 이러한 능력은 사용자의 특성이나 취향을 알아내고 그것에 따라서 사용자에게 가장 적절한 goal을 알아내어 수행할 수 있게 해주기 때문이다. 사용자의 행위를 보면서 사용자의 특성을 파악하는 문제는 귀납적 학습과 점진적 학습에

의해서 해결될 수가 있다. 즉, 사용자가 취하는 각각의 행동을 예제의 클래스에 속하게 되고 그러한 행동을 취하게 된 환경을 미리 설정된 속성들에 의해서 결정되어진다. 그러므로 사용자가 한 행동을 취하는 경우에 지능형 에이전트의 귀납적 학습모듈은 그때의 환경을 속성과 속성의 값으로 구성된 예제로 표현하고 그 예제의 클래스는 사용자가 취한 행동이 된다. 이와 같은 사용자의 행동은 점진적으로 일어나게 되므로 ID5와 같은 점진적 귀납적 학습방식은 사용자의 취향을 파악할 수 있게 된다. 경우에 따라서, 비점진적인 방식을 이용하여 사용자의 행동양식을 알아내고자 하는 경우는 ID3와 같은 프로그램이 이용되기도 한다 [22].

6. 맺음말

기계학습은 적응성이 있고 실수를 반복하지 않는 지능형시스템의 구축에는 필수적이다. 이와 같은 기계학습은 귀납적, 연역적, 사례기반 및 다전략방식과 같은 다양한 형태로 연구가 진행되고 있다. 본 논문에서는 이와 같은 기계학습에 대한 전체적인 개요를 설명하고 귀납적 기계학습의 대표적인 방식 중의 하나인 ID3계열에 대해서 예제를 중심으로 설명하였다. ID3계열의 기계학습은 비점진적방식과 점진적인 방식이 있으므로 각각의 경우에 대한 예제와 프로세스를 살펴보았다. ID3계열의 귀납적 기계학습은 무질서도를 최소화하는 방식을 재귀적으로 사용하여 결정트리를 구축하고 이와 같은 결정트리는 지식베이스로의 변환이 용이하다. 마지막으로 귀납적 기계학습이 지식습득문제를 극복할 두 방법을 제시하였고 또한 지능형 에이전트에서 사용자의 취향을 파악하는 용도로 이용될 수 있음을 서술하였다.

참 고 문 헌

[1] R. Bareiss, B. Porter and K. Murray. Supporting start-to-finish development of knowledge bases. *Machine Learning*, 4 : 259-283, 1989.

[2] J. G. Carbonell. An overview of machine learning. In *Machine Learning : An Artificial Intelligence Approach*, pages 3-16. Los Altos, CA : Morgan Kaufmann, 1983.

[3] W. Cohen. Abductive explanation based learning : A solution to the multiple explanation problem. Technical Report ML-TR-29, Laboratory for Computer Science Research, Rutgers University, New Jersey, 1989.

[4] A. Danyluk. Finding new rules for incomplete theories : Explicit biases for induction with contextual information. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 34-36, Ithaca, New York, June 1989.

[5] A. Danyluk. Generation and evaluation of contextual heuristics for inductive learning. Technical report, Columbia University, 1990.

[6] G. DeJong and R. Mooney. Explanation-based learning : An alternative view. *Machine Learning*, 1(2) : 145-176, 1986.

[7] T. G. Dietterich and B. G. Buchanan. The role of the critic in learning systems. Technical Report STAN-CS-81-891, Stanford University, CA, 1981.

[8] J. Gennari, P. Langley and D. Fisher. Models of incremental concept formation. *Artificial Intelligence*, 40 : 11-61, 1989.

[9] R Hall. Learning by failing to explain : Using partial explanations to learn in incomplete or intractable domains. *Machine Learning*, 3 : 45-77, 1988.

[10] K. J. Hammond. Explaining and repairing plans that fail. *Artificial Intelligence*, 45 : 173-228, 1990.

[11] R. M. Keller. Concept learning in context. *Machine Learning Journal*, pages 91-102, 1987.

[12] Y. Kodratoff and G. Tecuci. What is an explanation in DISCIPLE? In *Proceedings of the Fourth International Machine Learning Workshop*, pages 160-166, Irvine,CA, 1987.

[13] J. Kolodner. An introduction to case-based reasoning. Technical Report GIT-ICS-90/19, Georgia Institute of Technology, 1990.

[14] J. Kolodner and R. Thau. Design and implementation of a case memory. Technical Report GIT-ICS-88/34, Georgia Institute of Technology, 1988.

[15] J. E. Laired, A. Newell and P. S. Rosenbloom. Soar : An architecture for general intelligence. Artificial Intelligence., 33(1) : 1-64, 1987.

[16] J. E. Laired, P. S. Rosenbloom and A. Newell. Chunking in Soar : The anatomy of a general learning mechanism. *Machine Learning*, 1 : 11-46, 1986.

[17] R. Michalski, J. Carbonell and T. *Mitchell. Machine Learning : An Artificial Intelligence Approach : Volume I.* Tioga Publishing Company, Palo Alto, CA, 1983.

[18] R. Michalski, J. Carbonell and T. *Mitchell. Machine Learning : An Artificial Intelligence Approach : Volume II.* Morgan Kaufmann Publishers, Inc, CA, 1986.

[19] R. S. Michalski. A theory and methodology of inductive inference. In *Machine Learning : An Artificial Intelligence Approach*, pages 83-134. Los Altos, CA : Morgan Kaufmann, 1983.

[20] R. S. Michalski. Understading the nature of learning : Issues and research directions. In *Machine Learning : An Artificial Intelligence Approach Volume II*, pages 3-25, Los Altos, CA : Morgan Kaufmann, 1986.

[21] S. Minton, J. G. Carbonell, C. A. Knoblock, D. R. Kuokka, O. Etzioni and Y. Gil. Explanation-based learning : A problem solving perspective. *Artificial Intelligence*, 40, 1989.

[22] T. M. Mitchell, R. Caruana, D. Freitag, J. McDermott and D. Zabowski. Experience with a learning personal assistant. *Communications of the* ACM, 37(7) : 81-91, 1994.

[23] T. M. Mitchell, R. M. Keller, and S. T. Kedar-Cabelli. Explanation-based generalization : A unifying view. *Machine Learning*, 1(1) : 47-80, 1986.

[24] Utgoff P. ID5 : An incremental ID3. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 107-120, 1988.

[25] Utgoff P. Incremental induction of decision trees. *Machine Learning*, 4 : 161-186, 1989.

[26] Utgoff P. An improved algorithm for incremental induction of decision trees. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 318-325, 1994.

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

[28] Michalski R. and Tecuci G. *Machine Learning : A Multistrategy Approach : Volume IV.* Morgan Kaufmann Publishers, Inc., CA, 1984.

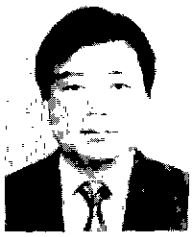
[29] H. A. Simon. Why should machines learn? In *Machine Learning : An Artificial Intelligence Approach*, pages 25-37. Los Altos,CA : Morgan Kaufmann, 1983.

[30] G. Tecuci and Y. Kodratoff. Apprenticeship learning in imperfect domain theories. In *Machine Learning : Artificial Intelligence Approach, Vol III.* Morgan Kaufmann, 1990.

[31] P. E. Utgoff. *Machine Learning of Inductive Bias.* Kluwer Academic Publishers, 1986.

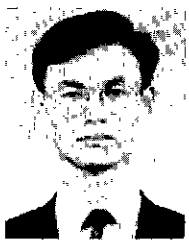
[32] Kodratoff Y. and *Michalski R Machine Learning : An Artificial Intelligence Approach : Volume III Morgan Kaufmann Publishers, Inc.*, CA,1990.

박 영 택



1978 서울대학교 전자공학과 학사
1980 한국과학기술원 전산학과 석사
1992 Univ. of Illinois at Urbana-Champaign 전산학과 박사
1981~현재 숭실대학교 컴퓨터학부, 부교수
관심분야 : 인공지능

이 강 로



1994 숭실대학교 전자계산학과 학사
1994~현재 숭실대학교 컴퓨터학부 대학원 재학중
관심분야 : 인공지능