□ 기술해설 □

적응형 에이전트

숭실대학교 백혜정*·김형훈**·이강건*·한경식**
이수원***·박영택***

● 목 차 ●

1. 서 론

지능형 에이전트는 사용자가 직접 수행하여야하는 각종 작업을 대신 수행해주는 소프트웨어이다. 사용자가 수행하고 싶은 작업은 경우에 따라서 복합적인 과정을 필요로 하기도 하고 단순히 하나의 타스크만을 수행하기도한다. 따라서 지능형 에이전트는 사용자가 요구하는 작업을 이해하고 이를 효과적으로 수행하기 위한 계획(Planning) 기능을 필요로하며 복잡한 작업을 효율적으로 수행하기 위해서 여러개의 에이전트가 협동으로 문제를 해결하기 위한 구조가 요구된다.

따라서 지능형 에이전트는 다중 에이전트가 서로 협동하여 가장 효과적으로 사용자의 요구를 충족시킬 수 있는 계획을 수립하고 이를 실현하는 소프트웨어이다.

이와 같은 에이전트가 보다 지능적이기 위해서는 에이전트를 사용하는 사용자에 적응할 수 있는 적응성(Adaptiveness)이 필요하다. 예를 들면, 웹을 탐색하는 에이전트는 현재 탐색을 명령한 사용자가 누구인가에 따라서 사용자의 취향에 따른 탐색 결과를 제공할 수 있어야한다. 또한, 전자 상거래에 이용되는 에이전트는 구매자의 취향에 가장 적절한 상품을 탐색하여 사용자의 만족도를 극대화할 수 있어야한다. 에이전트가 이러한 적응성을 가지지 못한다면 아무리 효과적으로 문제를 해결할 수 있다하더라도 사용자의 취향에 맞지 않는 결과들도 제공하므로써 사용자를 불편하게 만들게 되고 따라서 에이전트로서의 기능을 수행하지 못하게 된다.

따라서 지능형 에이전트는 당연히 사용자의 취향을 알아내고 이를 작업 수행에 적용할 수 있는 기능을 필요로한다.

이러한 적응성은 사용자 수준과 각 사용자마다의 타스크 수준에 따라서 가능해야한다. 이러한 기능은 인공지능의 기계학습(Machine Learning) 방식을 활용하여 많은 부분이 자동화될 수 있다. 현재, 지능형 에이전트의 연구들 중에는 많은 부분이 이러한 적응성을 증가시키기 위해서 기계학습의 다양한 방식을 활용하는 연구이다.

본 논문에서는 지능형 에이전트의 적응성을 증가시키기 위한 대표적인 연구를 조사하고 이를 분석하고자한다. 제2장에서는 적응형 에이전트에 필수적으로 이용되고 있는 기계학습에 대한 간단한 소개를 한다. 제3장은 기계학습을 활용하고 있는 인터페이스 에이전트, WebWatcher, Syskill&Webert, 동적환경에서의 에이전트에 대해서 기술하고 있다.

* 학생회원
** 비 회 원
*** 종신회원

2. 기계학습

기계학습(Machine Learning)이란 동일한 성격의 작업을 더욱 효과적이고 효율적으로 수행할 수 있도록 시스템을 적응적으로 변화시키는 것이라고 정의할 수 있다[1]. 가장 기초적인 기계학습 방법은 계산 결과를 단순히 기억하거나 교사(Instructor)로부터 주어진 정보를 지식베이스에 저장하여 다음 작업에 이를 효율적으로 이용하는 방법이다. 보다 복잡한 형태의 기계학습은 과거 문제해결(Problem Solving) 경험으로부터 새로운 기술(Skill)을 습득하거나, 또는 현재의 지식을 새롭게 구조화하는 것이다. 가장 상위단계의 기계학습은 새로운 개념을 습득하거나 새로운 지식을 발견하는 것이며, 기존의 경험을 토대로 하여 새로운 것을 창조해 내는 일도 포함된다. 이러한 여러 형태의 기계학습은 결과적으로 새로운 지식의 획득에 따른 지식베이스의 확장, 불필요한 지식의 삭제에 따른 지식베이스의 정제(Refinement), 휴리스틱의 습득에 따른 탐색속도의 개선 및 문제해결 결과의 개선 등의 효과를 얻게 되며, 궁극적으로 문제해결 범위를 확장하고 시스템의 성능을 향상시키게 된다.

기계학습은 1960년대부터 게임영역 등에서 연구되기 시작하였다[2]. 1980년대부터는 귀납적 학습(Inductive Learning) 및 연역적 학습(Deductive Learning)이 중점적으로 연구되어왔고 그 응용분야도 점차 확대되고 있다[3]. 귀납적 학습은 다수의 훈련사례(Training Example)로부터 일반적인 특성을 추출하여 새로운 개념을 습득하는 방법으로, 훈련사례가 긍정적(Positive) 또는 부정적(Negative)으로 분류되어 주어지는지의 여부에 따라 지도적 개념학습(Supervised Concept Learning) 및 개념군집화(Conceptual Clustering)로 세분된다.

지도적 개념학습에서는 각각의 훈련사례가 '속성-값(Attribute-Value)'의 쌍들로 표현되고 그 분류가 주어졌을 때, 어떠한 '속성-값'들이 만족되면 긍정적으로 분류되는가를 설명하는 개념을 습득하여 다음 사례가 어떻게 분류되는지를 예측한다. 지도적 개념학습은 결정트리(Decision Tree), 논리 등 개념을 표현하는 언어의 종류에 따라, 개념 공간(Concept Space)에서 특정한 개념을 학습결과로 선택하는 기준, 즉, 귀납적 편향(Inductive Bias)의 종류에 따라, 그리고 알고리즘이 전체 훈련사례들을 일괄적으로(Batch) 처리하는지 새로운 훈련사례가 주어졌을 때 이를 점진적으로(Incremental) 처리하는지에 따라 그 종류가 다양하다. 대표적인 지도적 개념학습방법으로는 ID3[4], AQ, 버전공간(Version Space) 등을 들 수 있다.

개념 군집화는 지도적 학습과는 달리 훈련사례가 외부로부터 분류되지 않은 채 주어진다. 시스템은 주어진 훈련사례 상호간의 유사성 및 이질성을 토대로 이를 군집화하여 새로운 사례가 어떤 군집에 속하는 지를 예측한다. 대표적인 개념 군집화방법으로는 통계적인 베이지안 추론방법을 사용하는 AUTOCLASS를 들 수 있다.

연역적 학습은 일반적으로 과거 문제해결(Problem Solving) 경험으로부터 분석적인 방법을 사용하여 지식을 습득하며, 습득된 지식은 새로운 유사한 문제를 해결하는데 길잡이로 적용되거나 영역지식을 더욱 효율적으로 적용하기 위한 탐색 제어 지식(Search Control Knowledge)을 생성하는데 이용된다. 그러므로, 연역적 학습에 사용되는 지식은 문제해결 경험 및 영역 지식(Domain Knowledge)이 가장 큰 비중을 차지하게 된다. 연역적 학습은 새로운 개념을 습득한다기보다는 기존의 개념에 대한 새로운 구조적인 기술(Structural Description)을 습득함으로써 시스템의 정확성이나 일반성을 저하시키지 않으면서 효율성을 향상시키는데 그 핵심이 있다. 따라서, 학습의 결과는 기존의 개념에 대한 새로운 영역지식의 습득이라는 측면에서 지식 습득, 그리고 시스템의 효율성 향상의 측면에서 기술 습득, 이 두 가지 관점에서 생각될 수 있다[5].

연역적 학습은 매크로 연산자(Macro-Operator) 이후 본격적으로 연구되기 시작하였으며, 최근에는 설명기반학습(Explanation-Based Learning), 및 청킹(Chunking) 등의 학습방법이 주 연구대상이 되고 있다. 매크로 연산자는 일련의 단순연산자(Primitive Operator)들에 의한 문제해결 과정을 하나의 복합연산자로

표현한 것으로서, 매크로 연산자를 습득하여 사용함으로써 단순 연산자를 탐색, 결정하는 과정을 줄일 수 있다. 설명기반학습은 하나의 긍정적 훈련사례(Positive Training Example)를 일반화(Generalization)시켜 목표개념(Goal Concept)에 대한 새로운 기술(Description)을 습득하는 학습방법이다. 청킹은 Soar라는 통합적 인공지능구조에 포함된 학습방법으로서, 설명기반학습과 거의 유사하나 Soar의 문제해결 메커니즘 및 규칙기반구조와 밀접하게 연관되어 있다는 데에 그 특징이 있다.

3. 적응형 에이전트 사례연구

대부분의 적응형 에이전트는 사용자와 상호작용을 하면서 사용자의 분야별 취향을 학습하고 이를 활용하는 용도로 귀납적 기계학습 방식을 이용하고 있다. 또한, 귀납적 기계학습에서 이용되고 있는 엔트로피 개념을 응용한 다양한 방식들이 연구되고 있다. 본 장에서는 여러가지 적응형 에이전트 중에서 대표적인 적응형 에이전트에 대해서 소개한다.

3.1 인터페이스 에이전트에서의 학습

인터페이스 에이전트란 인공지능 기법을 이용하여 특정한 응용 프로그램을 사용하는데 있어서 사용자에게 도움을 주는 프로그램이다. 인터페이스 에이전트를 구성하는 방법은 여러가지가 있다. 그 중 기계학습에 기반(Learning Based Approach)을 둔 방법은 최소한의 배경지식으로 사용자의 "행위"(Behavior) 을 학습함으로써 사용자를 돕는데 필요한 지식을 얻는 방법이다. 이 에이전트 구성 방법은 다른 방법들에 비해 많은 장점을 가지고 있다. 첫째, 이 방법은 사용자나 응용 프로그램 개발자로부터 많은 일을 요구하지 않고, 둘째, 에이전트는 매우 쉽게 사용자의 선호도나 습관에 적응할 수 있으며, 셋째, 여러 사용자의 정보, 습관, know-how 등이 다른 여러 에이전트사이에 교환될 수 있는 것을 들 수 있다. 기계학습에 기반을 둔 에이전트의 구성방법은 특정조건하에서 사용자에게 도움을 주는데 필요한 지식을 에이전트 스스로 습득할 수 있고, 스스로 지식을 프로그램 할 수 있다는 가정 하에서 출발한다. 일반적으로 에이전트는 (1)사용자 행위의 관찰을 통한 학습, (2)사용자의 피드백(Feedback)을 통한 학습, (3)훈련을 통한 학습, (4)다른 에이전트의 충고(Advice)를 통한 학습 등의 방법을 통하여 필요한 지식을 습득해 나간다(그림 1)[6].

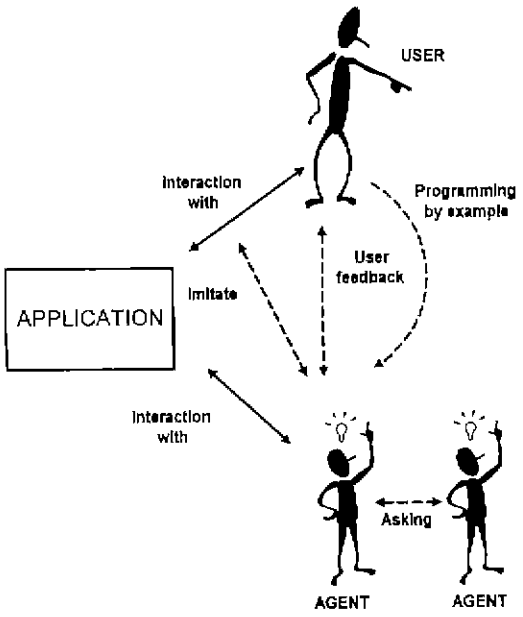

그림 1 인터페이스 에이전트에서의 학습

사용자 행위의 관찰을 통한 학습(Learning by Observing the User)방법은 에이전트가 사용자의 행위를 계속적으로 관찰하여 필요한 지식을 습득하고 학습하는 방식이다. 에이전트는 오랜 기간(몇 주 혹은 몇 달)동안 사용자의 행위를 모니터하고, 반복되는 행위의 패턴을 자동화한다. 예를 들어 사용자가 A로부터 오는 대부분의 메일은 읽고, B로부터 오는 대부분의 메일은 삭제한다면 에이전트는 어떠한 메일을 어떻게 처리해야 하는지에 대한 지식을 습득할 수 있다. 비슷한 예로서 뉴스 필터링 에이전트(News Filtering Agent)는 사용자가 주로 읽는 기사(Article)의 패턴을 학습하여 그와 비슷한 기사가 발견되면 사용자에게 그 기사를 제공할 수 있다. 이를 위한 대표적인 학습 방법은 기억기반 학습(Memory-Based Learning)을 사용한다. 사용자가 특정한 행동을 수

행할 때 에이전트는 사용자가 수행한 모든 상황-행위(Situation-Action)들을 메모리에 저장한다. 새로운 상황이 발생했을 때 에이전트는 그 상황을 저장된 상황-행위와 비교함으로써 가장 차이(Distance)가 작은 행위를 취하게 된다.

사용자의 피드백을 통한 학습(Learning From User Feedback)은 사용자의 직접적 또는 간접적인 피드백을 이용하는 학습방법이다. 간접적 피드백은 에이전트가 제의하는 행위를 사용자가 무시하고 다른 행위를 취할 때 일어난다. 피드백을 이용하는 한 가지 방법은 이 무시된 상황에 대한 정확한 행위를 새로운 예제로서 지식 베이스에 저장한다는 것이다. 또한 사용자는 "다음부터는 이런 행위를 하지마" 혹은 "나는 이 기사(Article)를 싫어해"와 같은 직접적 피드백을 줌으로써 에이전트의 행위를 바꿀 수 있다.

사례를 통한 학습(Learning by Being Trained)방식은 사용자가 의도적으로 사례를 제시하여 에이전트를 학습시키는 것이다. 사용자는 에이전트에게 가상의 사건, 상황에 대한 사례를 제시하고 그러한 상황에서는 무엇을 해야할 것인지를 보이면서 에이전트를 훈련시키는 것이다. 에이전트는 새로운 사례와 기존 사례 사이의 상관관계를 계산하고 사례 베이스를 적절히 변화시키면서 새로운 사례를 수용한다. 예를 들어 특정한 사람에게서 온 메일을 특정한 폴더에 저장하는 예를 보이면서 에이전트를 훈련시킬 수 있다. 이 기능은 가상 사례의 name field, folder field를 제외한 나머지 특징 field에 "Wildcard"를 사용함으로서 구현할 수 있다.

에이전트가 필요한 지식을 얻는 마지막 방법은 비슷한 일을 수행하는 다른 에이전트에게 충고를 요청하는 것이다. 에이전트는 어떤 특수한 상황에서 무엇을 해야할지 알지못한다면 다른 에이전트에게 그 상황을 제시하여 충고를 요청할 수 있다. 예를 들어, A로부터 어떤 메시지가 도착했고 그 메시지에 대해 어떤 행위를 취해야할지 모른다면 다른 에이전트에게 도움을 요청할 수 있다. 다른 에이전트들이 A로부터 온 메시지는 중요하고 사용자가 꼭 읽어야 한다고 충고한다면 에이전트는 그 충고를 받아들여 사용자에게 그 메시지를 읽도록 추천할 수 있다. 또한 에이전트는 다른 에이전트로부터 온 충고의 평균을 계산하기보다는 사용자가 지시한 특정 에이전트의 충고에 우선 순위를 둘 수 있다.

현재 활발히 연구 개발중인 인터페이스 에이전트는 전자 메일 에이전트, 뉴스 필터링 에이전트, 미팅 스케쥴링 에이전트, 서비스 선택 에이전트(Entertainment Selection Agent) 등이 있다.

3.2 WebWatcher

카네기 멜론 대학에서 만든 에이전트인 WebWatcher는 웹브라우저 상위에서 실행이 되면서, 웹에서 사용자가 원하는 정보를 검색할 수 있도록 도와주는 지능형 에이전트이다. 사용자가 웹에서 정보를 검색할 때, 각 웹페이지 내의 링크를 따라 정보를 검색하는데, WebWatcher는 사용자가 현재 보고 있는 페이지 내의 링크들중 가장 유력한 링크를 다음 검색 페이지로 추천한다. 이때 사용자는 WebWatcher가 추천하는 링크를 선택할 수도 있고, 사용자가 원하는 임의의 다른 링크를 선택 할 수도 있다. WebWatcher는 추천링크에 대한 사용자의 반응과 웹 링크를 통해 원하는 정보를 획득했는지의 성공 여부를 학습예제 집합으로 기록한다.

이렇게 형성된 학습예제 집합들에 기계학습을 적용하여 WebWatcher는 검색 제어 지식을 자동적으로 학습한다. 그래서, 시간이 지남에따라, 과거에 검색했거나, 다른 사용자가 검색했던 주제분야에 대한 검색 제어 지식을 학습하게 되어, 그 주제 분야의 웹 검색에는 전문가가 된다. 이러한 특징은 WebWatcher가 일종의 Learning Apprentice라는 것을 보여준다[7].

WebWatcher를 이용한 검색 과정은 다음과 같다. 먼저, WebWatcher 검색 에이전트의 페이지에 들어가면, 사용자는 찾고자하는 정보에 대해 간단하게 기술한 후 검색을 시작한다. 이 시점으로부터 WebWatcher는 사용자와 동반하여, 사용자의 행동을 관찰하고, 기존에 학습

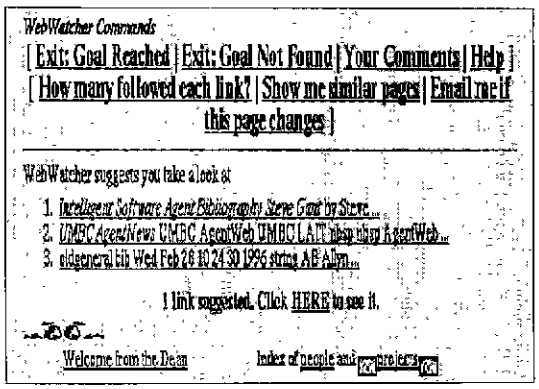

그림 2 WebWatcher

된 지식을 사용하여 다음 검색 링크를 제안하고, 사용자의 행적을 학습예제 데이타로 기록한다(그림 2).

WebWatcher는 사용자에게 원하는 정보를 얻을수 있는 유력한 링크를 제안하는데, 이것은 그림 2에서 눈모양으로 표시된 부분이다. WebWatcher는 사용자의 선택을 기다리는 동안, 학습을 통해 추천된 웹 페이지를 미리 분석(Prefetch)한다. 이때 사용자가 추천링크가 아닌 다른 새로운 링크를 선택하면, WebWatcher는 검색에 대한 기록을 수정하고, 새로운 웹 페이지를 검색, 분석한다. 또한, WebWatcher는 그림에서 보는것처럼 메뉴바를 두어, 여러 가지 사용자와 인터페이스를 한다. 특히 [Exit : Goal Reached, Goal Not Found]는 사용자가 원하는 정보를 얻었는지 여부를 표시하는 것이다. 이것이 선택되어지면, WebWatcher는 검색에 대한 기록 파일을 닫고, 사용자가 선택한 일련의 웹문서 링크들을 학습하게 된다. 그리고 WebWatcher 에이전트는 끝나게 된다. WebWatcher는 학습을 하여 검색의 효율을 향상시키는데, 이때 필요한 지식표현은 다음과 같다.

$$\text{LinkQuality} : \text{Page} * \text{Interest} * \text{Link} \rightarrow [0, 1]$$

이때, Page는 현재 웹 페이지를 의미하고, Interest는 사용자의 관심을 나타내는 단어 집합을 의미하고, Link는 웹페이지의 하이퍼링크를 의미한다. 이것은 임의의 사용자가 원하는 정보와 현재 페이지가 주어졌을 때 링크를 선택할 가능성으로 이함수의 값은 사용자가 링크를 선택할 것인지를 나타낸다. WebWatcher에서 학습은 사용자들로부터 얻은 학습예제집합에서 일반적인 링크값(LinkQuality)을 구하는 함수로 기존의 경험에서 학습하는 방식이다. WebWatcher의 학습 방법에는 크게 정보기반(Mutual Information) 기법을 이용한 방법과 TFIDF를 이용한 방법이 있다.

정보기반(Mutual Information) 기법을 이용한 방법은 먼저 사용자가 선택한 페이지에 연결되어있는 링크와 기존의 학습예제집합내에 존재하는 페이지의 링크와 유사도를 다음과 같이 측정한다.

$$I(D, L1, L2) = E(D, L1) - \frac{M_+}{M} * E(D_+, L1) - \frac{M_-}{M} * E(D_- + L1)$$

이때, D_+는 L2로의 링크를 포함하는 문서들(예제학습집합)의 집합을 의미하며, D-는 L2로의 링크를 포함하고 있지 않는 문서의 집합이다. 그리고 M은 해당하는 문서들의 개수이다. 이때, Webwatcher는 유사도가 높은 것을 사용자에게 연관된 페이지(Related Page)로 제한하고, 그중 가장 유사도가 높은 것은 다음 링크로 추천한다. 이 방법은 사용자의 관심, 현재 보고있는 페이지, 기존의 학습 예제 집합을 분석하고 유사도를 측정하여 링크여부값을 측정한다.

TFIDF방법은 현재 사용자의 관심과 하이퍼링크(Hyperlink)간의 유사도를 계산하는 방법

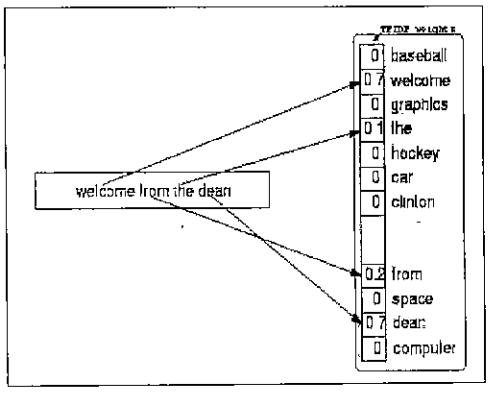

그림 3 TF-IDF

이다. TFIDF방법은 그림 3에서 보는것 처럼 단어를 벡터화하여 표시하는데, TF(Term Frequency)는 관심정보를 이끄는 정확한 예제집합에서 특정단어가 나오는 빈도를 나타내고 DF(Document Frequency)는 특정단어가 나오는 예제집합의 수를 의미한다. 그러므로, 단어의 중요도는 TF에 비례하고 DF에 반비례한다. 이때, 사용자의 관심을 나타내는 벡터와 기존의 학습된 링크의 양(사용자가 선택한 링크)의 벡터와 음(사용자가 선택하지 않은 링크)의 벡터간의 유사도를 계산하여 사용자 관심에 대한 링크값을 결정한다. 이렇게 WebWatcher는 예제학습집합을 학습함으로 검색제어 지식을 얻어, 사용자가 현재 검색하고 있는 페이지에서 가장 유력한 링크를 파악하여 추천한다[8, 9].

결국, WebWatcher는 웹브라우저 상위에서 실행하면서, 사용자의 관심 사항을 관찰하고, 경험들을 학습하여, 다음에 같은정보 검색이 들어오면 학습된 지식을 이용하여 가장 유력한 것을 추천하는 방식을 취하는 정보 검색 에이전트이다.

3.3 Syskill & Webert

Syskill & Webert[10]는 사용자의 관심에 대한 취향(Profile)을 학습하여 사용자가 관심을 가지는 웹 페이지를 구별해 내는 소프트웨어 에이전트이다. 이 에이전트가 동작하는 방식에는 두 가지가 있는데, 첫째는 인덱스 페이지의 일부 링크에 대해 사용자가 입력한 등급을 이용하여 나머지 링크들 중 사용자가 관심을 가지는 링크를 제시하는 것이다. 인덱스 페이지란 일반적으로 100개 이상의 다른 정보 자원으로의 링크를 포함하여 작성된 웹 페이지이다. 사용자 등급을 입력받기 위해, 사용자 페이지의 HTML 원문을 이용하여, 각 페이지별로 추가적인 기능을 부여한다(그림 4). 추가되는 기능으로는 해당 웹 페이지와 사용자의 관심도와의 연관성 등급(Hotlist, Lukewarm, Coldlist)[11] 입력, 인덱스 페이지로의 복귀 또는 주제 재설정, 현재 주제에 대한 명시적인 사용자 취향의 학습, 제안의 생성 또는 LYCOS 검색엔진 질의 등이 있다. Syskill & Webert가 동작하는

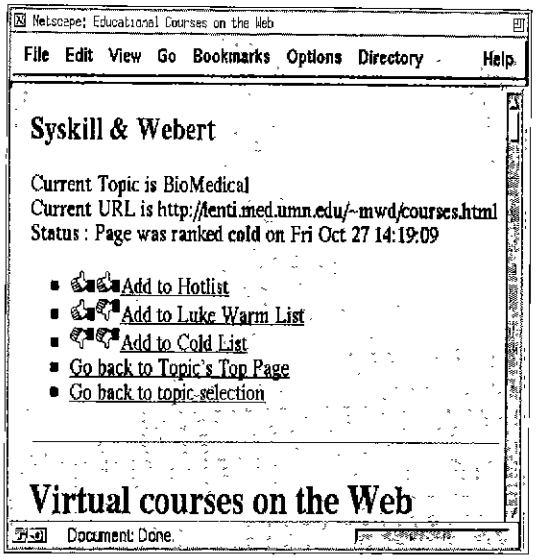

그림 4 등급설정을 위한 Syskill &Webert 인터페이스

두번째 방식은 사용자의 관심분야에 해당되는 웹 페이지들을 조회하는 LYCOS 질의를 작성하여, 질의 결과에 대해 주석을 부여하는 것이다. Syskill & Webert는 단일 페이지에 연결된 링크에 대한 제안을 생성하는 것만으로는 제한적일 수밖에 없다.

학습 알고리즘에는 주어진 개념에 대한 양의 예제(Positive Example)의 집합(사용자가 관심을 갖는 웹 페이지들)과 음의 예제(Negative Example)의 집합(사용자의 관심에 해당되지 않는 웹 페이지들)을 필요로 한다. 대부분의 학습기반 프로그램에서는 특징 벡터(Feature Vector)의 집합으로 표현되는 예제들을 요구하므로, Syskill & Webert에서는 웹 페이지의 HTML 원문을 이진 특징 벡터(Boolean Feature Vector)로 변환하는 방법을 구축하였다. 각 특징은 특정 웹 페이지에서 특정 단어의 존재 여부를 나타내는 이진 값을 포함한다.

HTML 문서에서 쓰인 모든 단어가 특징으로서 사용되는 것은 아니다. 따라서 어떤 단어를 특징으로서 선정하는지를 결정하기 위해 정보기반 접근방법(Information-Based Approach)을 사용한다. 직관적으로 볼 때, 사용자는 대체로 Hotlist(사용자가 관심을 갖는 웹 페이지 목록)에 포함되는 페이지에서는 자주, Coldlist

(사용자가 관심을 가지지 않는 웹 페이지 목록)에는 드물게 쓰이는 단어들을 선호한다. 이는 정보 획득의 기대치(E(W, S))를 산출하여 구할 수 있으며, 이는 단어(W)의 존재여부로써 웹 페이지 집합(S)의 원소들을 분류하는 것이다. 다음은 정보 획득의 기대치를 산출하는 공식이다.

$$E(W, S)=I(S)-[P(W=present)I(S_{w=present}) +P(W=absent)I(S_{w=absent})]$$

여기서 I(S)는 다음과 같이 산출된다.

$$I(S)=\sum_{c\in\{w,\ cold\}} -p(S_c)\log_2(p(S_c))$$

위 공식에서 P(W=present)는 W가 웹문서에 존재할 가능성이고, ($S_{w=absent}$)는 W가 최소한 하나 이상 존재하는 웹문서의 집합이며, S_c는 c부류(Class)에 속하는 웹문서들을 나타낸다. 이와 같은 접근방법을 사용하면 임의의 개수만큼 주요 단어를 추출해 낼 수 있다.

일단 HTML 원문을 특징 벡터로 표현되는 양, 또는 음의 예제들로 변환한 후에는 정보의 분류를 위한 확률적(Probabilistic) 방법론인 Bayesian 분류법, 저장 예제를 통한 분류법인 Nearest Neighbor, Nearest Neighbor에 가중치를 적용하여 변형한 PEBLS, ID3 등의 정보기반 기계학습법인 결정 트리(Decision Trees), 정보검색 분야에서 가장 성공적이고 많은 실험을 거친 기법인 TF-IDF 등과 같은 다양한 학습 알고리즘을 적용할 수 있다.

3.4 동적 환경에서의 적응형 에이전트

에이전트는 환경에 적절히 상호작용 할 수 있는 능력이 필요하다. 동적인 환경(Dynamic Environments)에서 에이전트는 지능적인 행위뿐만 아니라 반작용(Reactive)적인 행위도 할 수 있어야 하는데 이를 위해서는 다음과 같은 요소가 필요하다. 첫째, 계획(Planning)과 실행(Execution)이 밀접하게 결합된 복합구조(Hybrid Architecture)가 있어야 한다. 둘째, EBL과 같이 계획의 성능을 높여주거나 강화학습(Reinforcement Learning)과 같이 더 나은 실행을 하기 위한 학습방법이 필요하다. 셋째, 외부의 인터럽트에 적절히 대응할 수 있어야 한다. 이와 같이 동적인 환경에 적합한 에이전트를 구현할 수 있는 대표적인 에이전트 아키텍처로는 SEPIA[12]와 Soar[13]가 있다.

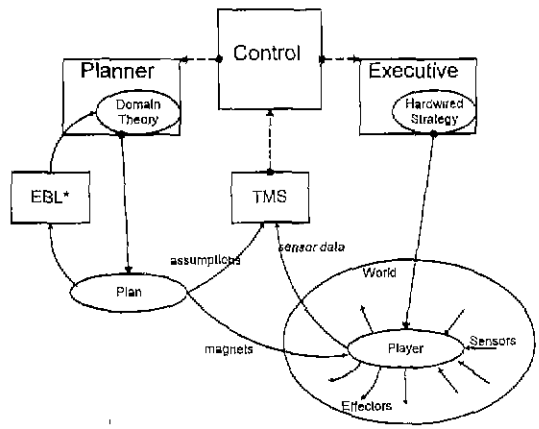

그림 5 SEPIA 아키텍처

SEPIA는 계획을 생성하는 계획기(Planner), 생성된 계획을 실행하는 실행기(Executive) 및 이 두 가지 모듈의 수행권을 제어하는 제어기(Control Module)로 구성되어 있다(그림 5). 실행기는 센서(Sensor)와 연결되어 외부의 변화를 감지할 수 있고, 입력된 외부 변화에 반사적으로 행위 할 수 있도록 미리 정의된 휴리스틱 정보가 내장되어 있다. 계획기는 에이전트가 취할 행위를 생성해 내고, 이는 한 단계씩 실행기에 의해 실행된다. 그러나, 외부의 급격한 변화에 즉각적인 대응이 필요할 때는 계산하는데 시간이 걸리는 계획기를 거치지 않고 실행기에 의해 미리 지정된 행위를 하게 된다. 이와 같이 계획기와 실행기 사이에서 수행권을 부여하는 것은 제어기가 하는 일이다. 이 세 가지 모듈 외에 EBL*모듈이 계획기와 함께 작동한다. EBL*모듈은 기존의 설명 기반 학습방법인 EBL에서 파생된 학습방법으로, 완전한 증명(Completed Proof)을 입력으로 받아 새로운 매크로 연산자(Macro-Operators)를 생성해 낸다. 이렇게 생성된 마크로 연산자는 탐색을 효율적으로 제어하여 문제해결의 속도를 증가시킨다. 따라서, 기존의 상황과 유사한 상황에서, EBL*는 매크로 연산자를 적용하

여 재계획 없이도 에이전트가 빠르게 행위 할 수 있도록 해준다. SEPIA는 TMS(Truth Maintenance System)를 사용하여, 외부의 변화와 계획 사이에 모순이 일어나면 인터럽트를 발생시키고 바뀐 부분에 대해 다시 계획을 한다.

또 다른 아키텍처로 통합적 인공지능 구조인 Soar가 있는데, 그 특성은 다음과 같다. 첫째, 계획과 실행은 같은 아키텍처와 지식베이스를 공유하므로 계획과 실행간에 지식을 전달하는 일이 필요 없다. 둘째, 행위는 매우 빠른 반작용(High-Speed Reflexe), 신중한 선택(Deliberate Selection), 제약 없는 계획과 문제해결(Unrestricted Planning and Problem Solving)의 세 단계에서 제어가 가능하다. 셋째, 학습은 계획하는 동안 지식을 습득하여 제어지식(Control Knowledge)의 형태로 생성규칙을 만들어 내고, 이렇게 생성된 규칙은 재계획할 필요 없이 빠르게 실행할 수 있도록 해준다. 이와 같이 Soar에서 계획, 실행, 학습은 매우 밀접하게 연결되어 있어서, 계획은 실행시에 필요에 따라 호출되며 계획하는 동안 얻어진 지식은 바로 학습된다. Soar에서는 기존 생성 시스템처럼 단일한(Monolithic) 계획을 실행하는 것이 아닌, 청킹(Chunking)에 의해 생성된 선호(Preference)가 포함된 제어규칙(Control Production)들을 실행한다. 이는 여러 가지 상황에서 얻은 제어규칙을 실행시에 상황에 따라 적절한 제어규칙만을 실행하여 다중계획의 효과를 나타낸다[14]. Soar에서는 EBL과 유사한 학습방법인 청킹을 사용하며, 청킹은 Soar의 문제해결 구조와 밀착 결합(Tightly Coupled)되어 있어서 실행 및 계획하는 동안 계속 학습을 한다. 또한, Soar는 계획하는 동안 외부에서의 간섭이나 외부 환경에 변화가 생겼을 경우 빠르게 대처할 수 있는데, 이는 계획 중에도 계속 외부로부터 입력을 받아들여 이를 일시적 기억장소(Working Memory)에 저장하고 이를 각 규칙의 전제조건(Precondition)과 부합(Matching)하기 때문에 가능하다. Soar를 이용하여 구현한 에이전트로는 RoboSoar가 있는데, RoboSoar가 하는 일은 카메라가 장착된 Puma robot arm을 이용하여 블록들을 작업에 맞춰 정렬하는 것이다. RoboSoar의 환경에서는 다른 에이전트에 의해 블록의 위치가 바뀌는 등의 간섭이 가능하고, 긴급 조명에 불이 들어왔을 경우 이를 우선적으로 처리할 수 있다. RoboSoar는 위에서 설명한 여러 Soar의 기능을 이용해서 지능적인 행위와 외부환경에 대한 반작용 등을 할 수 있다.

4. 결 론

지능형 에이전트의 중요한 기능은 사용자의 행위를 관측하면서 사용자의 타스크 단위의 취향을 추출하고 이를 이용하여 사용자에 적응할(adaptive) 수 있는 기능이다. 이러한 기능은 기존의 기계학습 방식을 활용하여 많은 부분이 자동화 될 수 있다. 본 논문에서는 이러한 적응형 에이전트를 위한 기계학습 방식에 대해서 설명하고 기존의 대표적인 적응형 에이전트 방식에 대해서 살펴보았다. 이와같은 적응형 에이전트에 대한 연구는 대부분이 귀납적인 방식을 이용하여 사용자의 행위를 내부 표현하고 이로 부터 사용자의 취향을 추출하고 활용하는 방식을 취하고 있다. 앞으로는 귀납적 방식이 추출한 사용자 취향을 보다 최적화하는 연역적 방식 및 혼합방식의 기계학습 방식이 많이 연구될 것으로 예상된다.

참고문헌

[1] Simon, H. A., "Why Should Machines Learn?". In Machine Learning : *An AI Approach,* Vol. 1, pp. 25-38, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, (eds.), Morgan Kaufmann, 1983.

[2] Samuel, A. L., "Some studies in machine learning using the game of checkers". In *Computers and Thoughts and Language,* E. A. Feigenbaum and J. Feldman (eds.), McGraw-Hill, 1963.

[3] J. Shavlik and T. Dietterich (eds.) *Readings in Machine Learning,* Morgan

Kaufmann, 1990.

[4] 이강로, 박영택, "ID3 계열의 귀납적 기계학습, 한국정보과학회, 제13권 5호, pp. 6-18, 1995.

[5] 이수원, "분석적 학습", 정보과학회지 13권 5호, pp. 33-44, 1995.

[6] P. Maes. Agent that reduce work and information overload. *Communications of the ACM,* Vol.37, No.7. 1994.

[7] R. Armstrong, D. Freitag, T. Joachims, T. Mitchell, "WebWatcher : A Learning Apprent .ice for the World Wide Web", *1995 AAAI Spring Symposium on Information Gathering from Heterogeneous,* Distributed Environments, March 1995.

[8] R. Armstrong, D. Freitag. T. Joachims, T. Mitchell, "WebWatcher : Machine Learning and Hypertext", *Fachgruppentreffen Maschinelles Lernen, Dortmund,* Germany, August 1995.

[9] R. Armstrong, D. Freitag, T. Joachims, T. Mitchell, "WebWatcher : A Tour Guide for the World Wide Web", *IJCAI97,* January 1997.

[10] M. Pazzani, J. Muramatsu, D. Billsus, "Syskill & Webert : Identifying Interesting Web Sites", *in Proceedings of the National Conference on Artificial Intelligence (AAAI96), Portland,* 1996.

[11] M. Pazzani, L. Nguyen, S. Mantik, "Learning from hotlists and coldlists : Towards a WWW information filtering and seeking agent", *http://www.ics.uci.edu/~pazzani/Coldlist.html.*

[12] Alberto Segre, and Jennifer Turney. Planning, Acting, and Learning in a Dynamic Domain. S. Minton(ed). *Machine Learning Methods For Planning.* Morgan Kaufmann Publishers. Inc, pp. 125-158, 1993.

[13] John Laird, and Paul Rosenbloom. Integrating Execution. Planning, and Learning in Soar for External Environments. In *Proceedings of the Eighth National Conference on Artificial Intelligence,* pp. 1022-1029, 1990.

[14] Soowon Lee, and Paul Rosenbloom. Granularity in Multi-Method Planning. In *Proceedings of AAAI-93,* pp. 486-491, 1993.

백 혜 정

1995 숭실대학교 전자계산학과 학사
1992~현재 숭실대학교 컴퓨터학부 대학원 재학중
관심분야 : 인공지능, 에이전트

김 형 훈

1993 숭실대학교 전자계산학과 학사
1996~현재 숭실대학교 컴퓨터학부 대학원 재학중
관심분야 : 인공지능, 에이전트

이 강 건

1996 안성산업대학교 전자계산학과 학사
1992~현재 숭실대학교 컴퓨터학부 대학원 재학중
관심분야 : Agent Modeling, AI Architecture, Planning, Machine Learning, Virtual Reality

한 경 식

1997 숭실대학교 컴퓨터 인공지능학과 학사
1996~현재 숭실대학교 컴퓨터학부 대학원 재학중
관심분야 : Agent, Planning, Machine Learing

이 수 원

1982 서울대학교 자연과학대학 계산통계학과 학사
1984 한국과학기술원 전산학과 석사
1984~1987 금성사 중앙연구소 주임연구원
1994 University of Southern California 전산학과 박사
1994~1995 USC/ISI 및 USC/IRIS 연구원
1995~현재 숭실대학교 컴퓨터학부 전임강사
관심분야 : AI, Planning, Machine Learning, Agent Modeling, Integrated AI Architecture

박 영 택

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

● 통신정보합동 학술대회 ●

- 일 자 : 1997년 4월 17~19일
- 장 소 : 부산 해운대
- 주 최 : 정보통신연구회
- 문 의 처 : 한국전자통신연구원 최문기 박사
  T. 042-860-6100

● 제24회 임시총회 및 춘계학술발표회 ●

- 일 자 : 1997년 4월 25(금)~26일(토)
- 장 소 : 한림대학교
- 발표논문 접수마감 : 1997년 3월 8일(토)
- 문의처 및 접수처 : 한국정보과학회 사무국
  T. 02-588-9246, F. 02-521-1352
  서울시 서초구 방배3동 984-1(머리재빌딩) ㉮137-063