Journal of Korea Multimedia Society Vol. 9, No. 7, July 2006(pp. 927-937)

MMORPG에서 결정트리 학습을 적용한 자동 프로그램 확인 기법

홍성우[†], 김형일[††], 김준태[†††]

요 약

자동 게임 프로그램(auto-playing game programs)은 게임 플레이어를 대신하여 게임 캐릭터를 조종하는 프로그램으로 MMORPG(massively multi-player online role playing game)에서 빈번히 사용되고 있다. MMORPG에서 게임 캐릭터의 레벨을 올리기 위해서는 경험치가 필요하며, 경험치 증가 과정에서 아이템을 구매할 때 사용되는 게임 머니와 특정한 기술을 사용할 수 있는 아이템을 획득한다. 이러한 레벨-업 과정에서 게임 플레이어들은 지루함을 느끼게 되고, 빠른 게임 캐릭터의 성장을 위해 자동 프로그램을 사용하여 게임 캐릭터의 레벨을 증가시키는 경우가 빈번히 발생한다. 그러나 자동 프로그램은 게임상에서 비정상적으로 자원을 독점하여 게임 시스템을 황폐화시킬 뿐만 아니라, 불법적인 수익사업으로 악용되어 건전한 게임산업 육성을 방해한다. 본 논문에서는 이러한 자동 게임 프로그램을 찾아내기 위하여 게임 캐릭터에 의해 발생되는 마우스와 키보드를 포함한 윈도우 이벤트 시퀀스를 분석하고, 이벤트 시퀀스로부터 속성 벡터를 생성하여 결정트리 학습을 수행하였다. 결정트리 학습은 윈도우 이벤트 시퀀스에 의해 생성된 속성 벡터들을 이용하여 자동 프로그램을 분류한다. 본 논문에서는 윈도우 이벤트 시퀀스를 활용하여 생성한 26개의 속성들을 결정트리 학습에 적용함으로써 MMORPG에서 자동 프로그램을 효과적으로 분류할 수 있다는 것을 MMORPG에 속하는 몇 가지 게임에 대한 실험을 통해 확인하였다.

Identification of Auto Programs by Using Decision Tree Learning for MMORPG

Sungwoo Hong[†], Hyungil Kim[††], Juntae Kim[†††]

ABSTRACT

Auto-playing programs are often used in behalf of human players in MMORPG(Massively Multi-player Online Role Playing Game). By playing automatically and continuously, it helps to speed up the game character's level-up process. However, the auto-playing programs, either software or hardware, do harm to games servers in various ways including abuse of resources. In this paper, we propose a way of detecting the auto programs by analyzing the window event sequences produced by the game players. In our proposed method, the event sequences are transformed into a set of attributes, and the Decision Tree learning is applied to classify the data represented by the set of attribute values into human or auto player. The results from experiments with several MMORPG show that the Decision Tree learning with proposed method can identify the auto-playing programs with high accuracy.

Key words: Machine Learning(기계학습), Decision Tree(결정트리), Auto Program(자동 프로그램), MMORPG

※ 교신저자(Corresponding Author) : 김형일, 주소 : 서울특별시 중구 필동 3가(100-715), 전화 : 02)2285-3712, FAX : 02)2285-3712, E-mail : hikim@dongguk.edu
접수일 : 2006년 5월 4일, 완료일 : 2006년 7월 21일
[†] 동국대학교 대학원 컴퓨터공학과
(E-mail : swhong@dongguk.edu)
[††] 정회원, 동국대학교 컴퓨터공학과 IT분야 교수요원
[†††] 동국대학교 컴퓨터공학과 교수
(E-mail : jkim@dongguk.edu)

1. 서 론

MMORPG(Massively Multi-player Online Role Playing Game)를 포함한 온라임 게임의 수요는 빠르게 증가하고 있다[3,5,15]. MMORPG에서 게임 캐릭터의 레벨을 올리기 위해서는 경험치가 필요하며, 경험치 증가 과정에서 아이템을 구매할 때 사용되는 게임 머니와 특정한 기술을 사용할 수 있는 아이템을 획득한다. 이러한 레벨-업 과정에서 게임 플레이어들은 지루함을 느끼게 되며, 빠른 게임 캐릭터의 성장을 위해 자동 프로그램을 수행하여 게임 캐릭터의 레벨을 증가시킨다. MMORPG에서는 게임 플레이어를 대신하여 캐릭터를 자동으로 제어하는 자동 프로그램은 소프트웨어 방식과 하드웨어 방식으로 나뉜다. 자동 프로그램은 게임 사용자를 대신해 캐릭터를 제어하여 게임을 자동으로 진행할 수 있는 특수 기능을 제공하는 프로그램(또는 매크로)이다[1]. 자동 프로그램은 키보드나 마우스에 이벤트를 발생시켜 게임상에서 게임 플레이어를 대신해 자동 사냥이나 자동 장사와 같은 행위를 지속적으로 수행시킨다.

동일한 반복 작업을 효율적으로 하는 것은 자동화(automation)의 중요한 목표이다. 온라인게임에서 자동화는 캐릭터 제어에 활용되고 있으며, 온라인게임에서 자동화를 가능하게 하는 것은 자동 프로그램이다. 그러나 이와 같은 편리한 기능의 자동 프로그램은 게임상에서 비정상적으로 자원을 독점하여 게임 시스템을 황폐화시킬 뿐만 아니라, 불법적인 수익사업으로 악용되어 건전한 게임산업 육성을 방해한다.

현재, 온라인게임 업체들은 자동 프로그램을 제지하기 위해 자동게임방지 프로그램을 사용하고 있지만, 이와 같은 업체의 노력에도 불구하고 소프트웨어 방식의 자동 프로그램은 여전히 패치를 수행하며 널리 활용되고 있다. 특히, 컴퓨터 본체와 외부 입출력 장치인 마우스, 키보드, 조이스틱, 모니터 사이에 위치하여 작동되는 하드웨어 방식의 자동 프로그램에 대해서는 기술적인 문제로 전혀 대응하지 못하는 실정이다.

MMORPG에서 자동 프로그램은 자동 사냥이나 자동 낚시 등과 같은 특정한 작업을 수행하기 위함을 목적으로 하며, 작업의 목적을 달성하기 위해 게임 캐릭터는 특정한 행위들을 반복 수행하게 된다. 게임 캐릭터가 발생시키는 윈도우 이벤트를 추출한다면 자동 프로그램에 의해 조종되는 게임 캐릭터를 선별할 수 있다. 또한 소프트웨어로 존재하든 하드웨어로 존재하든 자동 프로그램은 특정 작업을 수행할 때 활용되기 때문에 다양한 행위를 나타내는 플레이어 캐릭터와 자동 캐릭터는 구별될 수 있다. 즉, 특정한 작업을 수행할 때 발생시키는 행동 패턴을 자동 프로그램으로 조종되는 자동 캐릭터의 분류에 이용한다면 자동 프로그램 적용을 효과적으로 탐지할 수 있다.

본 논문에서는 자동 프로그램 탐지를 위해 게임 캐릭터에 의해 발생되는 마우스와 키보드를 포함한 윈도우 이벤트 시퀀스를 분석하고, 윈도우 이벤트 시퀀스를 속성 벡터로 변환하여 결정트리 학습에 적용한다. 결정트리 학습은 윈도우 이벤트 시퀀스에 의해 생성된 속성들을 이용하여 자동 프로그램 분류에 적용된다. 본 논문에서 생성한 이벤트 시퀀스를 활용한 26개의 속성들을 결정트리 학습에 적용하면 MMORPG에서 자동 프로그램을 효과적으로 분류할 수 있다.

2. 관련 연구

2.1 MMORPG과 자동 프로그램

MMORPG란 온라인게임과 RPG(role playing game)의 특성을 모두 소유한 장르로 온라인상에서 특정한 역할이 주어진 캐릭터를 이용하여 게임 플레이어간에 상호작용(의사소통, 협업, 경쟁, 등)을 발생시키며, 게임상에서 모험, 사냥, 임무 등을 행하는 PRG 기반 다중접속 온라인게임이다[9,15].

MMORPG에 존재하는 캐릭터는 크게 세 가지로 나뉠 수 있다. 게임 플레이어가 조종하는 플레이어 캐릭터인 HPC(human player character), 게임의 흥미를 부여하기 위해 게임 플레이어와 협업이나 경쟁을 수행하는 인공지능 캐릭터인 NPC(non-player character)이다. MMORPG에서는 게임 플레이어가 조종하는 게임 캐릭터를 자동 프로그램을 사용하여 조종하는 경우가 빈번하게 발생하며, 자동 프로그램에 의해 조종되는 자동 캐릭터를 APC(auto-player character)라 한다.

자동 프로그램은 게임 캐릭터의 레벨-업에 필요한 게임머니와 아이템 등을 빠르고 편리하게 획득하기 위해 게임 플레이어를 대신하여 게임 캐릭터를 조종한다. 자동 프로그램은 게임머니와 아이템 획득에 이용되는 사냥, 장사, 낚시 등을 게임 플레이어를

대신하여 수행할 수 있도록 제작된다. 자동 프로그램이 게임의 밸런스에 영향을 주어 게임 업체들은 자동게임방지 프로그램을 채택하여 자동 게임의 활용을 억제하지만, 자동 프로그램은 여전히 활용되고 있다. 자동게임방지 프로그램으로 소프트웨어 방식을 채택한 자동 프로그램의 활용이 위축되면서 하드웨어 방식의 자동 프로그램 기기가 등장하였다.

게임 업체들은 자동 프로그램을 해킹 프로그램으로 규정하여 게임 실행 이전에 방지하기도 하나, 자동 게임을 방지할 기술적 수준은 소프트웨어 방식의 자동 프로그램 정도만을 부분적으로 차단할 수 있는 수준에 머물러 있다.

하드웨어 방식의 자동 프로그램은 소프트웨어 방식의 자동 프로그램과 동작 방법이 상이하여, 현재 게임 업체들은 이와 같은 하드웨어 방식의 자동 프로그램 활용을 체크할 수 있는 기술은 전혀 완비되어 있지 못한 실정이다. 하드웨어 방식의 자동 프로그램은 컴퓨터 본체와 입력장치 중간에 위치하여 하드웨어를 기반으로 자동 게임을 지원하며, 하드웨어 방식의 자동 프로그램은 마우스와 키보드를 제어하기 위한 통신포트를 가지고 있다. 하드웨어 방식의 자동 프로그램도 소프트웨어 방식의 자동 프로그램과 동일하게 자동 사냥, 자동 장사 등에서 활용된다. 그림 1에 나타낸 그림은 하드웨어 방식의 자동 프로그램 기기이고, 그림 2에 나타낸 그림은 오토마우스라는 소프트웨어 방식의 자동 프로그램이다.

자동 프로그램을 방지하기 위해 사용되는 자동게임방지 프로그램은 게임에 사용되는 마우스 이벤트 정보를 가로챈 뒤 그에 따라 키보드나 마우스 이벤트를 발생시키는 소프트웨어 방식의 자동 프로그램을 차단한다. 그러나 하드웨어 방식의 자동 프로그램은 컴퓨터 외부의 독립적인 장치에서 키보드나 마우스 이벤트를 발생시키므로 자동게임방지 프로그램으로 검사하기 어렵다.

MMORP의 게임 사이즈가 커지고 게임 객체들간의 상호작용이 복잡해지면서 게임의 상위 수준 규칙들을 버그나 논리적인 에러 없이 프로그램 코드 안에 올바르게 구현하기가 점점 어려워지고 있으며, 이러한 버그를 게임 출시 전에 찾아내는 것은 어려운 문제이다. 그리고 게임 사용자의 속임수(cheat) 사용에 대한 방어책을 마련하는 것도 게임 개발에 있어서 중요한 문제이다. Delap 등[2]은 MMORPG와 같은

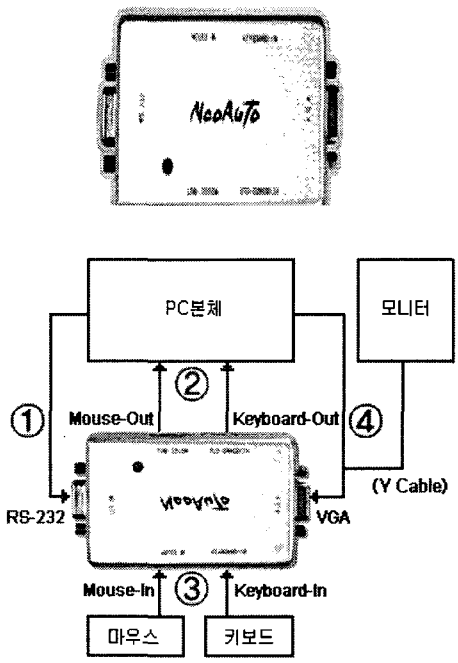


그림 1. 하드웨어 방식의 자동 프로그램 기기와 연결 구성도

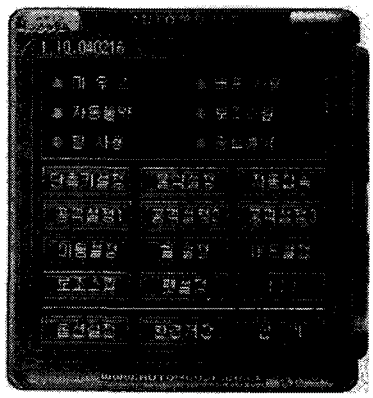


그림 2. 소프트웨어 방식의 자동 프로그램

대규모의 다중 사용자 게임에서 규칙기반이 적용된 실시간 검사 시스템을 활용하여 게임 규칙에 위배되는 경우를 탐지해냈다. 또한, 이와 같은 접근 방법으로 게임의 실시간 감시를 통해 프로그램의 버그를 찾아내고, 게임 사용자들의 부당한 속임수 사용을 탐지하였다. Delap 등은 게임 캐릭터가 획득한 게임 머니나 체력 등이 게임 규칙에 위배되는지를 검사함으로써 프로그램 버그나 속임수를 탐지하는 방법을 사용하였으며, 게임 규칙에 위배되는 판정을 위해 시간

간격 안에 획득할 수 있는 게임 머니와 체력들의 최대값을 게임 규칙의 임계값으로 활용하였다. 예를 들면, 게임의 실시간 검사에서 캐릭터의 게임머니나 체력이 게임 규칙의 임계값을 넘는 갑작스러운 증가가 발생할 경우 프로그램 버그나 게임 사용자의 속임수로 간주한다. 그러나 자동 프로그램은 게임 규칙을 위배하지 않은 상태에서 자동 게임을 수행하고 있기 때문에 이러한 방법으로는 자동 프로그램의 사용 여부를 감지할 수 없다.

2.2 결정트리

결정트리는 귀납적 방법의 학습 알고리즘으로 주어진 훈련 예제들(training examples)의 속성 값들을 기반으로 학습하여 트리를 생성하고 새로운 예제들을 분류하는데 활용하는 기법이다. 또한, 결정트리는 교사 학습(supervised learning)에 속하는 기계학습 기법으로 학습에 주어진 개체의 속성이 이산적인 값을 나타낼 경우에 효과적인 학습이 수행되며, 주로 분류 문제를 해결한다. 결정트리는 분류 모델 생성이 용이하며, 결정된 분류 모델은 이해하기 쉽다는 장점이 있다.

학습 결과인 트리는 노드(node)와 가지(branch)로 구성되며, 단말노드(leaf node)는 분류 클래스(class)를 의미하고, 루트(root)와 중간노드(intermediate node)들은 속성에 대한 검사를 표시한다. 또한 각 노드 사이의 가지(branch)는 검사의 결과를 의미한다. 새로운 사례의 분류는 결정트리의 루트에서 단말노드까지의 경로에 따라 사례 속성 값들을 검사하는 방식으로 이루어진다.

본 논문에서 자동 프로그램 분류를 위해 C4.5를 채택한 이유는 캐릭터에서 발생되는 윈도우 이벤트가 연속적인 속성값을 포함하기 때문이다. ID3는 연속적인 속성값을 적용할 수 없지만, C4.5는 연속적인 속성값을 적용한 학습이 가능하다. 기계학습에 많이 활용되는 베이지안 네트워크나 신경망 등은 연속적인 속성값에 대한 적용이 어려우며, 속성값 종류의 증가에 따라 학습 수행시간 또한 기하급수적으로 증가된다는 단점이 있다. 그러나 C4.5의 경우 이산적인 속성값을 포함하여 연속적인 속성값을 학습에 사용할 수 있으며, 연속적인 속성값을 이산적으로 변환한 후에 학습을 수행하기 때문에 학습 수행시간이 다른 기계학습 기법에 비해 빠른 학습을 수행할 수 있다. MMORPG가 실시간 온라인게임인 것과 캐릭터의 이벤트 발생이 이산적인 것과 연속적인 것이 동시 발생한다는 것을 고려하면 캐릭터가 발생시키는 윈도우 이벤트 분석을 통한 자동 프로그램 감지에는 C4.5가 효과적인 기계학습 기법이다.

본 논문에서는 C4.5의 특성으로 윈도우 이벤트 시퀀스 분석 도구로 C4.5를 채택하였다. 이벤트 시퀀스를 분석하여 분류 모델을 생성하기 위해 WEKA에서 제공하는 결정트리를 사용하였으며, WEKA에서 사용한 결정트리는 J48로 C4.5를 자바로 구현한 것이다. WEKA는 Waikato 대학에서 구현한 기계학습(machine learning) 라이브러리로 자바로 구현한 여러 가지 기계학습 알고리즘과 전처리(data pre-processing), 평가도구(evaluation methods)를 제공한다.

결정트리는 각 속성들의 정보획득량(information gain)을 계산하여 트리를 생성한다. 예를 들면, 클래스가 C_1, C_2 · · ·, C_n 인 훈련집합(training set) T가 있을 경우 평균 정보량(information or entropy)은 식 (1)과 같이 계산된다. 속성값이 V_1, V_2 · · ·, V_v 인 서로 다른 속성값을 갖는 속성 A_k 의 정보획득량은 공식 2와 같이 계산된다. 공식 (1)과 공식 (2)만을 적용하여 결정트리를 구성하는 것은 ID3이다.

$$Info(T) = \sum_{i=1}^{n} - P(C_i) log\, P(C_i) \quad (1)$$

$$Gain(A_k) = Info(T) - \sum_{i=1}^{v} \frac{|T_i|}{|T|} Info(T_i) \quad (2)$$

ID3가 이산적인 속성값을 갖는 경우에서는 상당히 좋은 결과를 낼 수 있지만, 속성값의 종류가 많은 속성을 보다 중요시하는 편향적인 성향을 보인다. 예를 들어 데이터 집합에 10개의 클래스가 존재하고 a1 속성이 10가지의 속성값을 가질 경우 다른 속성은 사용되지 않고 a1 속성으로만 모든 데이터를 분류하는 경우가 발생할 수 있다. 이런 현상은 과적합(over fitting) 문제를 발생시킬 수 있다. ID3의 이러한 문제를 C4.5에서는 속성값 종류의 수량에 따라 정보획득량을 일반화함으로 ID3에서 발생할 수 있는 편향성이나 과적합 문제를 교정하였다. 이를 위해 C4.5에서는 속성값 V_1, V_2 · · ·, V_v 을 이용하여 속성 A_k의 분할 정보량(split information)을 공식 (3)에 의해 계산한 후, 공식 (4)와 같이 속성 A_k의 정보획득율(gain ratio)을 이용하여 속성의 중요도를 결정하고 트리를 구성하는 방식을 취한다.

$$Split\quad Info(A_k) = \sum_{i=1}^{v} - P(V_i)\log P(V_i) \tag{3}$$

$$Gain\ \ Ratio(A_k) = Gain(A_k) / Split\quad Info(A_k) \tag{4}$$

기계학습의 활용 분야는 대단히 넓으며, 그 중에 결정트리는 모델 생성이 쉽고, 이해하기 쉬어 높은 활용도를 나타낸다. 결정트리 학습은 분서 분류[10,17], 시스템에 불법적으로 접근하는 침입 탐지[4,6], 단백질이나 질병 분류[7,14], 이미지 처리를 위한 속성 선택[8,13,18], 음성인식을 위한 음색 분류[11,12,16] 등을 포함한 분류나 탐지 문제 등에 다양하게 응용된다.

3. 이벤트 시퀀스 분석

본 장에서는 본 논문에서 제안한 자동 프로그램 분류 방법에 대해 기술한다.

소프트웨어 방식과 하드웨어 방식의 자동 프로그램에 의해 조종되는 자동 캐릭터와 플레이어 캐릭터의 다양한 윈도우 이벤트들을 추출하고, 자동 프로그램 분류를 위해 추출한 윈도우 이벤트를 이용하여 다양한 이벤트 시퀀스를 생성하였다.

게임이 진행되는 동안 키보드와 마우스를 포함한 다양한 윈도우 이벤트들이 발생된다. 표 1에 윈도우 이벤트에 대한 간단한 예를 표현하였다. 이벤트의 종류(action)는 키 누름, 마우스 버튼 클릭, 마우스 움직임, 등과 같은 게임을 수행할 때 발생할 수 있는 이벤트들의 종류를 나타내며, 메시지(message)는 이벤트들에 대한 각각의 윈도우 메시지를 나타낸다. 키보드 메시지에 대한 매개변수(parameter)는 각 키에 대한 식별자로 특정한 값을 나타내고, 마우스 메시지에 대한 매개변수는 X-Y에 대응되는 좌표값을 나타낸다. 표 2는 뮤 게임에서 자동 프로그램에 의해 자동 캐릭터가 몬스터 사냥을 수행하는 동안 발생된 이벤트 시퀀스에 대한 예제를 나타낸다.

다양하게 발생되는 이벤트 시퀀스들 사이의 직접적인 비교는 매우 어렵다. 그래서 가능한 특성화된 이벤트 시퀀스를 다양하게 추출하여 이벤트 시퀀스에 대한 분석을 통해 플레이어 캐릭터와 자동 캐릭터를 효과적으로 분류할 수 있는 속성들을 생성하였다. 속성들은 각 이벤트 사이들의 시간에 대한 평균, 표준편차, 특정 이벤트의 발생율, 이벤트 패턴 등을 포

표 1. 윈도우 이벤트

Action	Message	Parameter H	Parameter L
left alt	WM_SYSKEYUP	56	14354
space	WM_KEYUP	57	14624
caps lock	WM_KEYUP	58	14868
F1	WM_KEYUP	59	15216
F2	WM_KEYUP	60	15473
:	:	:	:
Begin mouse event	WM_KEYUP	0	229
Mouse left button down	WM_LBUTTONDOWN	X	Y
Mouse left button up	WM_LBUTTONUP	X	Y
:	:	:	:

표 2. 이벤트 시퀀스 예제

Message	Parameter H	Parameter L	Time
WM_KEYDOWN	57	14624	1540664
WM_KEYUP	57	14624	1540782
WM_KEYDOWN	57	14624	1541076
WM_KEYUP	57	14624	1541186
:	:	:	:
WM_KEYUP	2	561	1541476
WM_KEYUP	0	229	1541546
WM_LBUTTONDOWN	322	503	1541546
WM_LBUTTONUP	322	503	1541716
:	:	:	:

함한다. 자동 프로그램을 탐지하기 위해 생성한 속성들의 종류와 설명을 표 3에 나타내었다.

플레이어 캐릭터와 자동 캐릭터로부터 획득된 이벤트 시퀀스들은 속성을 이용하여 26차원 벡터로 변환되어 학습 데이터로 활용된다. 그림 3은 뮤 게임에서 자동 프로그램으로 게임을 진행하는 동안 1분 간격으로 이벤트 시퀀스를 처리하여 10개의 속성 벡터들을 생성한 예제를 나타낸다.

그림 3에 나타낸 속성 벡터는 플레이어 캐릭터와 자동 캐릭터에 의해 발생되는 이벤트 시퀀스들로부터 생성되며, 속성 벡터들의 집합은 결정트리 학습에 적용된다. 다양한 게임의 특성이나 자동 프로그램의 많은 게임 규칙들을 생성한 속성들의 조합에 의해 플레이어 캐릭터와 자동 캐릭터를 분류할 수 있다.

속성을 생성하기 위해서 먼저 게임 진행상에서 발생하는 이벤트들을 추출하였다. 게임 진행 시에 발생되는 키보드와 마우스의 입력 이벤트를 추출하기 위해서 "Tasker"라는 매크로 프로그램을 사용하였다. Tasker는 마우스의 움직임과 키보드의 이벤트 발생을 밀리초 단위로 파일에 기록한다.

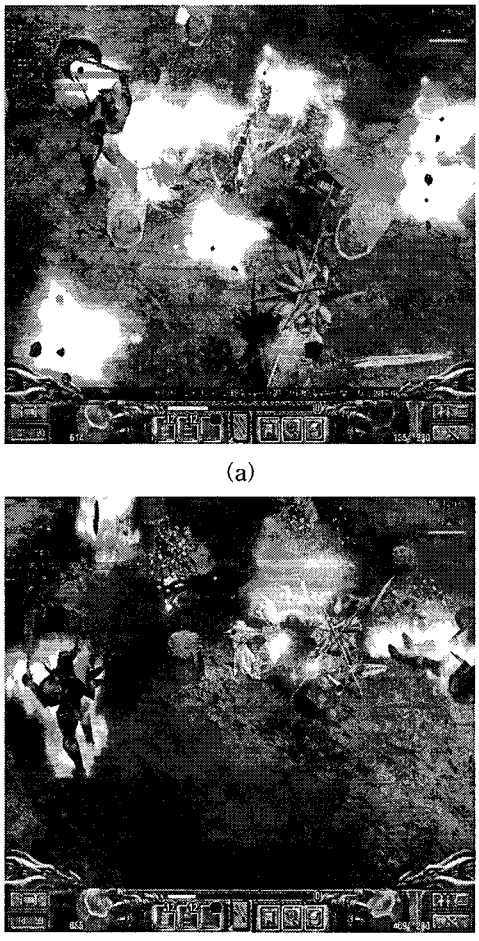
그림 4는 뮤(Mu) 게임에서 이벤트 추출을 위해 게임을 진행하는 모습을 나타낸 그림이다. 뮤 게임에서 추출한 이벤트는 자동 프로그램으로 진행한 90분 분량과 게임 플레이어가 게임을 수행한 90분 분량으로 총 180분 분량이다. 그리고 결정 트리에 사용할 이벤트 시퀀스 개체의 속성을 추출하기 위해 1분 단

표 3. 이벤트 시퀀스를 이용하여 생성한 속성 리스트

Attribute	Explanation
total_event(=a1)	마우스와 키보드를 포함한 이벤트들의 총수
key_event_ratio(=a2)	전체 이벤트에 대한 키보드 이벤트의 비율
autokey_ratio(=a3)	이벤트 총 수에 대한 자동 프로그램용 키의 비율
m_stroke_ave(=a4)	마우스 버튼을 눌렀을 때부터 뗄 때까지의 평균 시간
k_stroke_ave(=a5)	키보드를 눌렀을 때부터 뗄때까지의 평균 시간
ml_cont_ave(=a6)	연속하여 마우스 왼쪽 버튼 이벤트가 발생했을 때 이 이벤트들 사이의 시간에 대한 평균
mr_cont_ave(=a7)	연속하여 마우스 오른쪽 버튼 이벤트가 발생했을 때 이 이벤트들 사이의 시간에 대한 평균
sp_cont_ave(=a8)	연속하여 스페이스바 이벤트가 발생했을 때 이 이벤트들 사이의 시간에 대한 평균
autokey_cont_ave(=a9)	연속하여 자동 프로그램에서 사용되는 키가 발생했을 때 이 이벤트들 사이의 시간에 대한 평균
ml_kind_ave(=a10)	마우스 왼쪽 버튼 이벤트들 사이의 시간에 대한 평균
mr_kind_ave(=a11)	마우스 오른쪽 버튼 이벤트들 사이의 시간에 대한 평균
sp_kind_ave(=a12)	스페이스바 이벤트들 사이의 시간에 대한 평균
autokey_kind_ave(=a13)	자동 프로그램에서 사용되는 키들 사이의 시간에 대한 평균
m_stroke_dev(=a14)	버튼을 눌렀을 때부터 뗄 때까지의 시간에 대한 표준편차
k_stroke_dev(=a15)	키보드를 눌렀을 때부터 뗄 때까지의 시간에 대한 표준편차
ml_cont_dev(=a16)	연속하여 마우스 왼쪽 버튼 이벤트가 발생했을 때 이 이벤트들 사이의 시간에 대한 표준편차
mr_cont_dev(=a17)	연속하여 마우스 오른쪽 버튼 이벤트가 발생했을 때 이 이벤트들 사이의 시간에 대한 표준편차
sp_cont_dev(=a18)	연속하여 스페이스바 이벤트가 발생했을 때 이 이벤트들 사이의 시간에 대한 표준편차
autokey_cont_dev(=a19)	연속하여 자동 프로그램에서 사용되는 키가 발생했을 때 이 이벤트들 사이의 시간에 대한 표준편차
ml_kind_dev(=a20)	마우스 왼쪽 버튼 이벤트들 사이의 시간에 대한 표준편차
mr_kind_dev(=a21)	마우스 오른쪽 버튼 이벤트들 사이의 시간에 대한 표준편차
sp_kind_dev(=a22)	스페이스바 이벤트들 사이의 시간에 대한 표준편차
autokey_kind_dev(=a23)	자동 프로그램에서 사용되는 키들 사이의 시간에 대한 표준편차
ml_pattern(=a24)	마우스 왼쪽 버튼에 대한 시계열 패턴 값
mr_pattern(=a25)	마우스 오른쪽 버튼에 대한 시계열 패턴 값
sp_pattern(=a26)	스페이스바에 대한 시계열 패턴 값

109,	0.40,	0.05,	0 ,	0.55,	0.29,	0 ,	... ,	80.17,	3.92,	0,	0,	0,	0,	A
53,	0.42,	0.04,	0 ,	0.55,	0.32,	0 ,	... ,	5.08,	4.22,	0,	0,	0,	0,	A
105,	0.41,	0.08,	0.02,	0.50,	0.28,	0.02,	... ,	5.01,	3.99,	0,	0,	0,	0,	A
152,	0.43,	0.07,	0.01,	0.49,	0.28,	0.03,	... ,	5.01,	3.72,	0,	0,	0,	0,	A
104,	0.45,	0.05,	0 ,	0.50,	0.33,	0.02,	... ,	5.06,	3.78,	0,	0,	0,	0,	A
100,	0.41,	0.05,	0 ,	0.54,	0.27,	0.03,	... ,	5.01,	4.20,	0,	0,	0,	0,	A
100,	0.42,	0.05,	0 ,	0.53,	0.27,	0.03,	... ,	4.95,	4.17,	0,	0,	0,	0,	A
108,	0.43,	0.06,	0 ,	0.52,	0.31,	0.01,	... ,	5.00,	4.16,	0,	0,	0,	0,	A
94,	0.35,	0.10,	0.04,	0.51,	0.22,	0.02,	... ,	4.93,	4.19,	0,	0,	0,	0,	A
60,	0.48,	0.13,	0 ,	0.38,	0.25,	0 ,	... ,	126.15,	3.68,	0,	0,	0,	0,	A

그림 3. 샘플 데이터에서의 속성 벡터들



(a)

(b)

그림 4. 뮤에서 이벤트 추출을 위한 게임 진행 : (a) 자동 프로그램을 이용한 사냥, (b) 게임 플레이어의 사냥

위로 이벤트 데이터들을 재배열한 후, 속성을 추출하였다. 그러므로 뮤 게임에서 생성한 이벤트 시퀀스 개체의 총수는 180개이다. 이벤트 시퀀스 추출에 사용한 게임은 뮤(Mu), 아크로드(Archlord), 군주(GoonZu), 메이플스토리(Maple Story) 이고, 생성한 이벤트 시퀀스의 수량은 뮤에서 추출한 이벤트 시퀀스의 개체 총수와 동일하다. 그러나 메이플스토리는 마우스를 사용하지 않는 게임으로 키보드만을 활용하여 게임을 진행하기 때문에 마우스를 활용한 이벤트 데이터는 추출하지 않았으며, 마우스를 이용한 속성들도 생성하지 않았다.

4. 실 험

본 실험을 수행하기 위해 현재 인터넷에서 서비스되고 있는 MMORPG에 속하는 게임들 중에 "뮤(Mu), 아크로드(Archlord), 군주(GoonZu), 메이플스토리(Maple Story)"를 이용하여 실험하였으며, 실험은 자동 프로그램의 동작 방식에 따라 크게 두 가지로 나뉜다. 하나는 하드웨어 방식의 자동 프로그램에 대한 실험으로 실험에 사용된 게임은 뮤와 메이플스토리 이고, 다른 하나는 소프트웨어 방식의 자동 프로그램에 대한 실험으로 실험에 사용된 게임은 아크로드와 군주이다.

각 게임은 3명의 게임 플레이어가 90분씩 게임을 진행하고, 1분 간격으로 이벤트 시퀀스를 생성하였다. 자동 프로그램 또한 90분 동안 동일한 상황에서 게임을 진행시키고, 1분 간격으로 이벤트 시퀀스를 90개 생성하였다. 게임 플레이어와 자동 게임 모두 이벤트 생성시 단순한 마우스 움직임은 배제하였다. 메이플스토리는 마우스를 사용하지 않고 키보드만을 이용하는 게임이기 때문에 이벤트 시퀀스 생성시 키보드만을 이용하였다. 생성된 이벤트 시퀀스들을 이용하여 속성 벡터로 변환하였고, 훈련집합의 전체 데이터 수량은 각 게임마다 180개이다. 실험은 10회

교차 검증(10-fold cross validation)으로 수행하여 정확도를 측정하였다.

각 게임에서 정보획득량이 높은 상위 7개 속성들의 정보획득량을 표 4에 나타내었다. 본 결과를 보면 생성된 속성들은 각 게임에서 자동 게임의 분류를 위해 매우 유용하게 활용되는 것을 알 수 있다.

뮤 게임의 경우 가장 효과적인 속성은 연속한 마우스 오른쪽 버튼의 이벤트 발생 시간에 대한 평균과 표준편차인 a07과 a17이다. 만약 평균시간이 0.37초보다 작은 평균시간이라면 자동 프로그램을 97.2%의 정확도로 분류할 수 있으며, 시간의 표준편차 0.23초 보다 작다면 자동 프로그램을 95.6% 정확도로 분류할 수 있다. 뮤 게임에서 a07과 a17이 가장 잘 분류하는 이유는 뮤에서 몬스터 사냥을 수행할 때 마우스 오른쪽 버튼을 활용하는 경우에 몬스터에게 높은 데미지를 줄 수 있는 기능이 포함되어 사냥 시 활용도가 가장 높은 버튼이기 때문이다. 기본적으로 게임 플레이어는 자동 프로그램보다 천천히 게임을 수행하고, 연속한 마우스 버튼의 시간에 대한 표준편차가 게임 플레이어가 자동 프로그램보다 크다.

속성 a01은 전체 이벤트의 총수로 아크로드를 제외한 모든 게임에서 정보획득량이 높은 상위 7개 속성에 포함된다. 뮤에서 a01이 173보다 클 경우 자동 프로그램을 91.7%의 정확도로 분류할 수 있다. 이와 같은 이유는 게임 플레이어는 일반적으로 자동 프로그램에 비해 천천히 게임을 진행하기 때문이며, 다른 게임 플레이어의 방해도 발생될 수 있기 때문이다. 또한 자동 프로그램은 상황 인식의 부재로 동일한 행위를 중복 발생시킬 수 있다.

표 5은 각 게임에서 전체 속성을 이용한 분류 정확도를 나타낸 것이다. 모든 경우, 본 논문에서 제안한 자동 프로그램 분류 방법이 93% 이상의 정확도를 나타낸다. 그림 5부터 그림 8까지는 각 게임에서 C4.5 결정트리 알고리즘에 의해 생성된 결정트리를 나타낸다.

표 4. 개별 속성들의 정보획득량

Hardware type auto program				Software type auto program			
Mu		Maple Story		Archlord		GoonZu	
Attr.	Gain	Attr.	Gain	Attr.	Gain	Attr.	Gain
a07	0.926	a23	0.745	a22	0.778	a06	0.711
a17	0.820	a13	0.732	a08	0.740	a10	0.684
a01	0.789	a09	0.684	a11	0.687	a01	0.640
a11	0.670	a05	0.656	a06	0.599	a04	0.570
a04	0.659	a03	0.651	a04	0.417	a14	0.506
a14	0.628	a19	0.606	a03	0.416	a20	0.485
a13	0.648	a01	0.555	a02	0.412	a24	0.356

표 5. 전체 속성을 이용한 정확도

Hardware type auto program				Software type auto program			
Mu		Maple Story		Archlord		GoonZu	
Attr.	Acc.	Attr.	Acc.	Attr.	Acc.	Attr.	Acc.
all	99.4%	all	95.1%	all	93.9%	all	95.9%

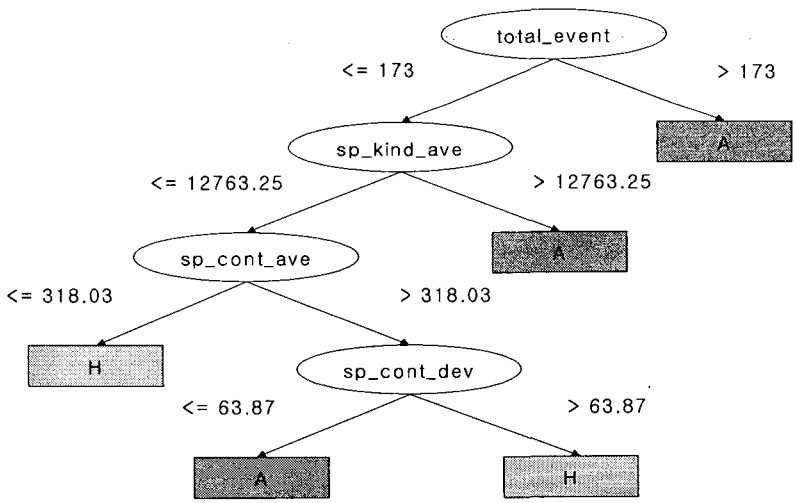


그림 5. 뮤 게임에서의 결정트리(정확도 = 99.4%)

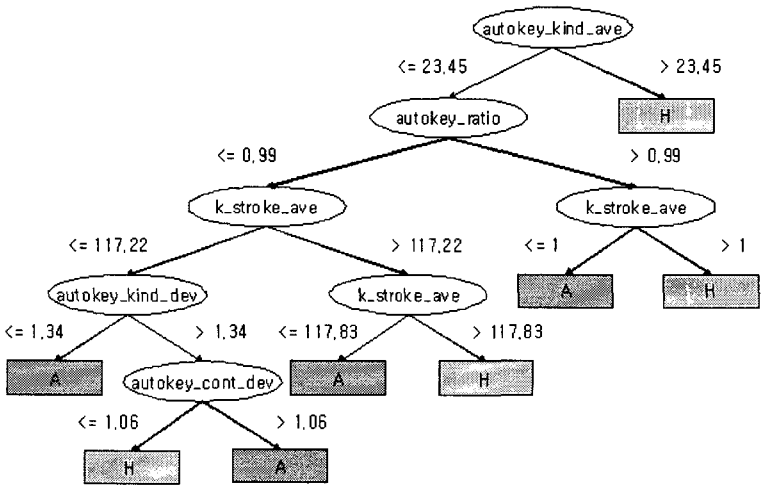


그림 6. 메이플 스토리에서의 결정트리(정확도 = 95.1%)

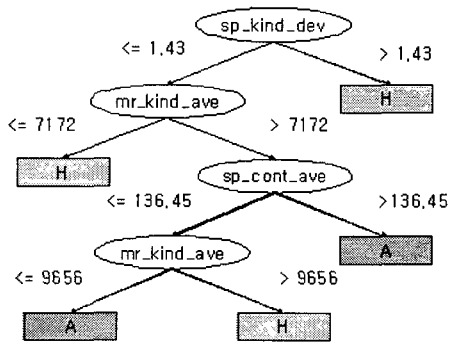


그림 7. 아크로드 게임에서의 결정트리(정확도 = 93.9%)

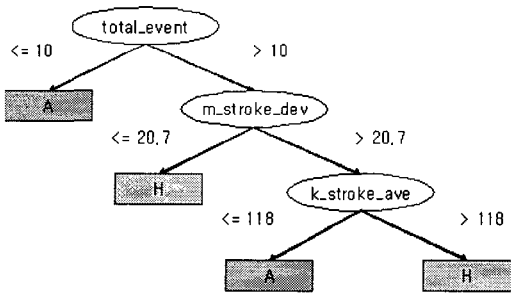


그림 8. 군주 게임에서의 결정트리(정확도 = 95.9%)

일반적으로 자동 프로그램은 게임 플레이어보다 더 많은 이벤트를 발생시킨다. 첫 번째 이유는 자동 프로그램은 게임 플레이어보다 기술 사용이 평균적으로 빠르기 때문이며, 두 번째 이유는 많은 경우에 있어서 자동 프로그램은 게임 상황을 인식하지 못하기 때문에 특정 임무를 수행하기 위해 반복적인 이벤트를 발생시키기 때문이다. 또한 자동 프로그램은 게임 플레이어보다 더 많은 이벤트를 발생시키지만 기계적인 이벤트 발생으로 인해 게임 플레이어보다 연속한 동일 이벤트 발생 시간에 대한 편차가 작다. 그러므로 이벤트 시퀀스의 특성에 따라 속성들의 조합을 사용한 결정트리 학습은 자동 프로그램을 높은 정확도로 탐지할 수 있다.

추가적으로 결정트리 이외의 기계학습 기법을 이용하여 동일한 실험 방법으로 여러 기계학습 기법들의 분류 성능을 비교 실험하였다. 표 6에 다른 기계학습 기법들의 분류 정확도를 나타내었다. 뮤 게임과 메이플 스토리 게임에서는 결정트리가 높은 정확도를

표 6. 다른 기계학습 기법들의 분류 정확도

Classifier	Hardware type auto program		Software type auto program		Average
	Mu	Maple Story	Archlord	GoonZu	
Decision Tree	**99.4%**	**95.1%**	93.9%	95.9%	96.1%
Multilayer Perceptron	96.1%	91.2%	**98.0%**	97.5%	95.7%
k-Nearest Neighbor	97.2%	90.2%	95.9%	**97.5%**	95.2%
Nave Bayesian	95.0%	93.1%	96.6%	91.9%	94.2%

나타내었고, 아크로드 게임에 대한 실험에서는 다중 퍼셉트론 기법이 가장 우수한 결과를 나타내었으며, 군주 게임에서는 k-NN(k-Nearest Neighbor)이 가장 우수한 성능을 나타내었다. 평균 정확도는 모든 경우에 대해 비슷한 성능을 나타내었지만, 결정트리 학습이 다른 기계학습들에 비해 자동 프로그램을 분류하는데 가장 우수한 분류 정확도를 나타내었다.

5. 결 론

자동 프로그램은 캐릭터를 사용자 대신 조종하여 게임을 진행하는 프로그램이다. 자동 프로그램은 다양한 기능을 가지고 있으며, 하드웨어 방식으로까지 발전하고 있다. 자동 프로그램은 게임상에서 비정상적으로 자원을 독점하여 게임 시스템을 황폐화시킬 뿐만 아니라, 불법적인 수익사업으로 악용되어 건전한 게임산업 육성을 방해한다.

본 논문에서는 게임이 진행될 때 발생되는 캐릭터의 윈도우 이벤트 시퀀스를 분석하고, 이벤트 시퀀스로부터 생성된 속성들을 이용하여 결정트리 학습을 통해 자동 프로그램 탐지 방법을 제안한다.

이벤트 시퀀스 생성을 위해 먼저 플레이어 캐릭터와 자동 프로그램으로 조종되는 자동 캐릭터의 윈도우 이벤트를 추출하였다. 이벤트들의 사이에서 평균 시간이나 시간의 표준편차, 이벤트 패턴 등을 이용하여 다양한 속성들을 생성하였다. 생성된 이벤트 시퀀스의 속성들은 26차원 벡터로 재표현 되고, 결정트리 학습에 훈련 데이터로 이용된다.

일반적으로 자동 프로그램은 게임 플레이어보다 많은 이벤트를 발생시키고 동일한 종류의 이벤트 사이의 평균 시간이나 표준편차 등이 게임 플레이어보다 작았다. 4개의 서로 다른 온라인게임에서 본 논문에서 제안한 방법이 자동 프로그램을 높은 정확도로 탐지할 수 있다는 것을 실험을 통해 확인하였다.

References

[1] A. Cypher, D. C. Halbert, D. Kurlander, H. Lieberman, D. Maulsby, B. A. Myers, and A. Turransky, *Watch what I do: programming by demonstration*, MIT Press, 1993.

[2] M. DeLap, B. Knutsson, H. Lu, O. Sokolsky, U. Sammapun, I. Lee, and C. Tsarouchis, "Is runtime verification applicable to cheat detection?," *The 3rd ACM Workshop on Network and System Support for Games*, 2004.

[3] N. Ducheneaut and R. J. Moore, "The social side of gaming: a study of interaction patterns in a massively multiplayer online game," *In Proceedings of the ACM conference on Computer-Supported Cooperative Work*, 2004.

[4] D. K. Kang, D. Fuller, and V. Honavar, "Learning Classifiers for Misuse and Anomaly Detection Using a Bag of System Calls Representation," *Proceedings of 6th IEEE Systems Man and Cybernetics Information Assurance Workshop*, 2005.

[5] J. Krikke, "South Korea Beats the World in Broadband Gaming," *IEEE MultiMedia*, 2003.

[6] W. Lee and S. J. Stolfo, "A framework for constructing features and models for intrusion detection systems," *ACM Transactions on Information and System Security*, 2000.

[7] M. K. Markey, G. D. Tourassi, and C. E. Floyd "Decision tree classification of proteins identified by mass spectrometry of blood serum samples from people with and without lung cancer," *Proteomics*, 2003.

[8] I. Park, K. Lee, and S. Lee, "Perceptual grouping of 3-D features in aerial image using decision tree classifier," *IEEE International Conference on Image Processing*, 1999.

[9] G. Robert, Pierre Portier, and Agnes Guillot, "Classifier systems as 'Animat' architectures for action selection in MMORPG," *In Game-On 02 : The Third International Conference on Intelligent Games and Simulation*, 2002.

[10] S. Sakurai and A. Suyama, "Rule discovery from textual data based on key phrase patterns," *Proceedings of the 2004 ACM symposium on Applied computing*, 2004.

[11] A. Samouelian, "Use of inductive learning for speech processing," *Australian and New Zealand Conference on Intelligent Information*

System, 1996.

[12] T. Shinozaki and S. Furui, "Error analysis using decision trees in spontaneous presentation speech recognition," *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2001.

[13] M. Simard, S. S. Saatchi, and G. D. Grandi, "The use of decision tree and multiscale texture for classification of JERS-1 SAR data over tropical forest," *IEEE Transactions on Geoscience and Remote Sensing*, 2000.

[14] A. Stasis, E.Loukis, S. Pavlopoulos, and D. Koutsouris, "A Decision Tree-Based Method, Using Auscultation Findings, for the Differential Diagnosis of Aortic Stenosis from Mitral Regurgitation," *Computers in Cardiology*, 2003.

[15] S. L. Whang and G. Y. Chang, "Lifestyles of Virtual World Residents, Living in the on-line game, "Lineage," *International Conference on CYBERWORLDS*, 2003.

[16] P. Wong and M. Siu, "Decision Tree Based Tone Modeling for Chinese Speech Recognition," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2004.

[17] W. Zheng, E. Milios, and C. Watters, "Filtering for Medical News Items Using a Machine Learning Approach," *Symposium of American Medical Informatics Association*, 2002.

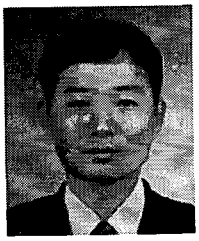
[18] Y. Yuan, Q. Song, and J. Shen, "Automatic video classification using decision tree method," *Proceedings of 2002 International Conference on Machine Learning and Cybernetics*, 2002.

[19] Mu, http://www.muonline.co.kr/

[20] Archlord, http://archlord.naver.com/

[21] GoonZu, http://www.goonzu.com/

[22] MapleStory, http://maplestory.nexon.com/



홍 성 우

2004년 동국대학교 대학원 컴퓨터공학과(공학석사).
관심분야 : 게임, 기계학습, 지능형 에이전트

김 형 일

1996년 목원대학교 수학과 졸업(이학사).
1996년~1998년 (주)경기은행.
2001년 동국대학교 대학원 컴퓨터공학과(공학석사).
2004년 동국대학교 대학원 컴퓨터공학과(공학박사).
2005년~2006년 동국대학교 컴퓨터공학과 IT분야 교수요원(정보통신부).
관심분야 : 게임, 지능형 에이전트, 기계학습, 정보검색, e-learning

김 준 태

1986년 서울대학교 제어계측공학과 졸업(공학사).
1990년 미국 Univ. of Southern California, Electrical Engineering-Systems(M.S.).
1993년 미국 Univ. of Southern California, Computer Engineering(Ph.D.).
1994년~1995년 미국 Southern Methodist University, Computer Science and Engineering(Postdoc).
1995년~현재 동국대학교 컴퓨터공학과 교수.
관심분야 : 기계학습, 데이터마이닝, 정보검색, 지능형 에이전트, e-learning