# Estimation-Exploration 알고리즘과 진화 신경망을 이용한 능동적인 데이터 수집 및 상대 플레이어 모델링

박현수 김경중*

세종대학교 컴퓨터공학과

hspark@sju.ac.kr, kimkj@sejong.ac.kr

# Active Data Collection and Opponent Player Modeling using Estimation-Exploration Algorithm and Evolutionary Neural Network

Hyunsoo Park Kyung-Joong Kim*

Dept. of Computer Engineering, Sejong Univ.

**요 약**

많은 게임에서 상대 플레이어의 전략 및 행동을 예측하여 그에 적합하게 대응하는 것은 게임의 결과를 결정하는 핵심적인 요소이다. 상대 플레이어의 행동을 예측하는 방법 중 하나는 다량의 게임 플레이 데이터를 수집한 뒤, 모델링하고, 이 모델을 상대 플레이어의 행동 예측에 사용하는 것이지만, 경우에 따라서는 한정된 게임 횟수만으로 플레이어를 모델링 해야 할 필요가 있다. 본 논문에서는 제한된 데이터 수집 횟수만이 주어졌을 때, Estimate-Exploration 알고리즘을 이용하여 데이터 수집을 목적으로 능동적으로 게임을 플레이하며, 상대 플레이어의 플레이 방식을 모델링 할 것을 제안한다. 반복 죄수의 딜레마 게임을 대상으로 하여 제안하는 방법을 실험하고, 그 결과를 무작위로 수집된 데이터를 이용하여 기존의 기계학습 알고리즘(C4.5, Multi-Layer Perceptron)을 이용한 결과와 비교한다. 그 결과 제안하는 방법은 데이터가 부족할 경우에 상대적으로 좋은 성능을 보이는 경향을 확인할 수 있었다.

## 1. 서론

일반적인 게임에서 플레이어는 이득을 최대화 시키기 위해 주어진 상황에서 최선의 선택을 한다. 그러나, 정보가 불확실할 수록 최선의 선택을 하는 것이 힘든 경우가 많으며, 특히, 다른 플레이어의 행동은 쉽게 예측할 수 없다. 때문에, 플레이어의 행동을 모델링하고 예측하는 많은 연구들이 있다[1][2].

상대방을 모델링하기 위한 대표적인 방법은 대량의 게임 플레이 데이터와 다양한 기계학습 기법을 이용하는 것이다. 그러나, 만약 기존의 방법을 이용하여 특정 플레이어의 단기적인 의도를 인식하고 적합한 대응을 하도록 해야 한다면, 단시간에 특정 플레이어의 플레이 데이터를 대량으로 수집해야 하지만, 이것은 현실적으로 불가능에 가깝다.

그리고, 데이터가 지나치게 부족하다면 기존 방법은 과적합(overfitting) 문제가 발생할 수 있다. 과적합 문제가 발생하는 이유는 수집된 데이터가 전체 데이터 분포를 대표하지 못하기 때문이다. 만약 수집된 데이터가 적더라도, 전체를 충분히 대표 할 수 있다면 과적합 문제를 완화 시킬 수 있는 가능성이 있다.

본 논문에서는 과적합 문제를 억제 할 수 있도록 능동적으로 데이터를 수집할 수 있는 방법으로서 Estimate-Exploration Algorithm (EEA)을 이용할 것을 제안한다. EEA는 역공학을 위해 제안된 알고리즘으로서 내부 구조를 알 수 없는 복잡한 시스템을 모델링하기 위해 제안되었다[3]. 데이터 수집, 다양한 가설 제시, 가설간의 불일치 발견, 불일치 제거를 위한 데이터 수집과 같은 네 단계 과정을 반복함으로써, 이미 수집된 데이터를 이용해 효율적으로 데이터를 수집하고, 시스템을 모델링 한다. 이 알고리즘의 핵심 아이디어는 주어진 데이터를 설명할 수 있는 다양한 가설(모델)간에 가장 일치하지 않는 설명이 가장 불분명한 부분이며, 다음 데이터 수집 행위는 이 부분을 확실히 할 수 있도록 수행되어야 한다는 것이다.

본 논문에서는 간단한 게임에서 인공지능 플레이어가 상대 플레이어를 모델링하기 위한 목적으로 게임을 진행하며, 효율적으로 데이터를 수집하고 상대방을 모델링하기 위한 방법으로 EEA를 사용하는 것을 보인다. 제안하는 방법을 실험하기 위한 게임으로 반복 죄수의 딜레마(Iterated Prisoner's Dilemma, IPD)를 이용하였다[4]. 이 게임은 상대 플레이어의 행동을 예측하는 것이 매우 중요하여 논문의 주제에 적합하다.

기존 연구[5]에서도 상대 플레이어를 모델링하기 위한 목적으로 EEA를 사용하였다. 그러나, [5]에서는 상대 플레이어를 표현력과 데이터의 축약 능력이 부족한 테이블(Lookup Table)로 가정하였으며, 실험에 이용한 상대 플레이어의 전략도 제한적이었다. 본 논문에서는 [5]를 개선하여 더 유연한 구조를 가질 수

있는 진화 신경망(Evolutionary Neural Networks)을 이용하였고, 유명한 전략 중 다섯 가지를 선정하여 실험을 진행하였다.

성능의 비교를 위해 EEA를 이용하지 않고 데이터를 무작위로 수집하여 기존의 기계학습 알고리즘을 이용하여 학습한 모델과 성능을 비교하였다. 그 결과, 수집된 데이터가 적을 때 대부분의 경우 기존 방법에 비해서 높은 성능을 보여줬다.

## 2. 제안하는 방법

### 2.1 Estimate-Exploration Algorithm

이 알고리즘의 크게 네 단계로 나눠진다. 1) 실험수행, 2) 다양한 가설 생성, 3) 최대 불일치점 탐색 그리고 4) 다음 실험 계획, 이 과정을 반복한다. 실험단계에서 계획된 실험을 수행하여 하나의 데이터를 수집하는 과정이다. 여기서는 상대 플레이어와 실제 게임을 플레이하는 것을 의미한다. 최초 게임에서는 무작위로 계획된 게임을 진행한다.

가설 생성 단계에서는 수집된 데이터를 설명할 수 있는 다양한 모델을 생성한다. 총 *N*개의 모델(본 논문에서는 15개)을 생성한 뒤, 성능이 나쁜 *N/2*개를 삭제한다. 여기서는 진화 신경망을 이용 했다. 신경망의 구조(은닉계층과 노드의 개수) 및 연결 가중치를 진화연산(Genetic Algorithm)을 이용하여 학습했다. 은닉계층은 0-5개, 한 계층에 노드는 1-10개 사이로 제한했다. 신경망의 구조에 따라서 모델이 가질 수 있는 표현력이 다르기 때문에 각각의 모델(가설)은 관측되지 않은 데이터에 대해서 서로 다른 결론을 내릴 수 있다. 추가로 신경망의 연결에 비용을 부가하여 관측된 데이터를 설명할 수 있는 가장 간단한 신경망을 생성하도록 유도하였다. 신경망의 연결에 비용이 추가될 경우 불필요한 연결이 줄어들어 구조가 단순해 지고, 빠른 학습이 가능해진다고 알려져 있다[6].

최대 불일치점을 탐색하는 문제는 일종의 최대값 탐색 문제로서, 진화 연산을 이용했다. 여기서 탐색 공간은 플레이 가능한 모든 게임구성을 의미한다. 그 중에 현재 생성된 모델간에 결론, 즉, 각각의 모델이 예측하는 상대 플레이어의 다음 행동이 가장 다른 게임구성을 찾아내는 것이다. 그러나, 자신의 행동은 스스로 조작 가능하지만, 상대방의 행동은 원하는 대로 조작할 수 없어 원하는 데이터에 대한 탐색이 실패할 가능성도 있다. 따라서, 새로운 데이터를 이용하여 기대 불일치를 제거하는 것을 최소로 하는 것이 적합하다. 특정 게임구성$j$ 탐색을 성공했을 때, 제거될 불일치를 $d_j$ 라고 하고, 탐색을 시도한 횟수를 $t_j$, 탐색이 성공한 횟수를 $s_j$ 라고 한다면, $j$ 를 탐색했을 때 기대 가능한 제거 불일치는 아래와 같다. 기대 불일치를 최대로 하는 게임 구성 $j$ 을 다음 게임 계획으로 선택한다.

$$D_j = d_j \times \frac{s_j}{t_j}$$

EEA는 언제나 새로운 데이터 수집 계획을 세우기 위해 여러 개의 가설을 생성하고 있다. 상대 플레이어의 미래 행동을 예측할 때, 이것을 일종의 앙상블로 이용하여 하여, 다수결로 예측을 수행했다.

이 외의 기본적인 구조는 [5]와 동일하다.

### 2.2 반복 죄수의 딜레마(Iterated Prisoner's Dilemma)

반복 죄수의 딜레마[4]는 두 명의 플레이어가 협력(Cooperation, C)과 배신(Defection, D)을 선택하는 단순한 게임으로써, 서로의 선택에 따라 보상을 받는다. 자신의 선택뿐만 아니라, 상대방의 선택에 따라 받는 보상이 다르기 때문에, 상대방의 전략에 적합하게 대응하는 것이 필요하다.

## 3. 실험 및 결과

본 논문에서는 상대 플레이어는 게임에 참가하는 두 플레이어의 과거 다섯 행동(C 또는 D)을 기억하고, 이 기억을 이용하여 다음(여섯 번째)행동을 결정한다고 가정하고, EEA를 이용하여 (여섯 번째) 행동을 예측할 수 있는 모델을 생성하는 실험을 한다. 이것은, 일반적인 관점에서 10개의 속성을 입력 받아 두 개(C, D)의 클래스 중 하나로 분류하는 문제로 볼 수 있다.

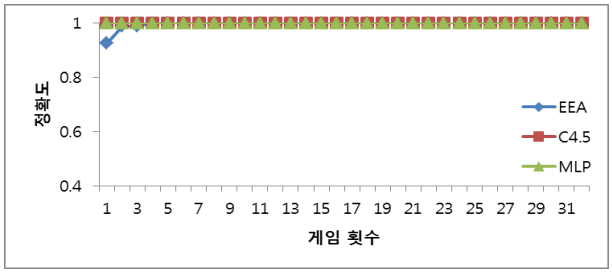
표.1 상대 플레이어의 전략

| 이름 | 설명 |
|---|---|
| AllC | 언제나 협력 |
| TFT | 상대방의 이전 행동을 따라 함 |
| NTFT | 10% 확률로 행동이 바뀌는 TFT |
| Major | 상대방이 가장 많이 한 행동을 따라 함 |
| Pav | 초기에 협력, 그 뒤로는 바로 이전 상대방의 행동과 자신의 행동이 같을 때만 협력 |

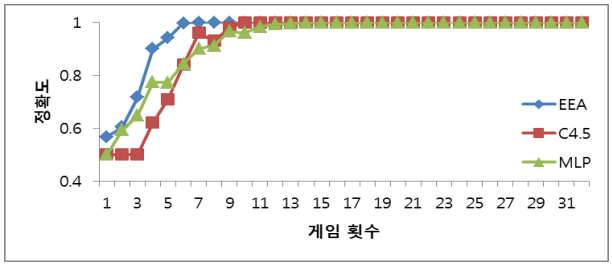
두 플레이어의 과거 10개의 행동만을 고려하기 때문에 한 게임이 구성 가능한 모든 경우의 수는 총 $2^{10}$=1024개이다. 그 중 특정 전략에 따라 가능한 최소의 게임 구성은 $2^5$=32개이다. 만약 상대 플레이어가 10개이상 과거의 행동을 고려하거나, 무작위적인 요소를 가지고 있다면, 그 이상의 구성이 가능하다. 본 논문에서는 대표적인 전략[7] 중 다섯 가지에 대해서 실험을 진행하였다. 실험에 사용한 전략은 표. 1에 소개하였다.

게임 횟수에 따른 정확도의 변화는 그림 1과 같다. 30번 실험을 반복하여 그 평균을 나타낸 것이다. 한번 게임 할 때마다, 하나의 데이터를 수집하기 때문에 더 적은 게임에서 높은 성능이 우수한 결과이다. 총 가능한 게임구성 32개를 테스트 데이터로 사용했다.

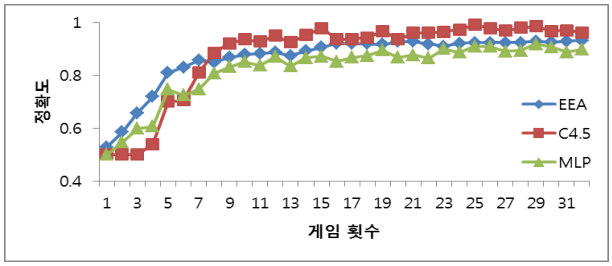
비교를 위해 WEKA[8]의 C4.5와 MLP를 이용했다. 능동적으로 계획을 세우고, 수집하는 EEA와는 달리, 무작위로 추출된 동일한 숫자의 데이터를 이용하여 학습한 결과를 비교했다.



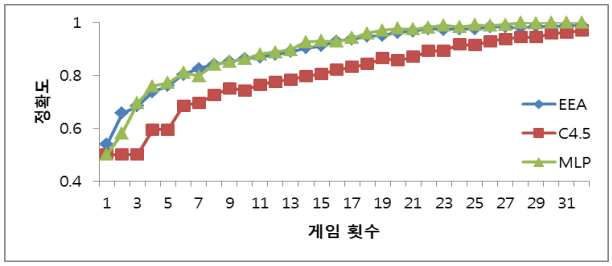

(a) AllC



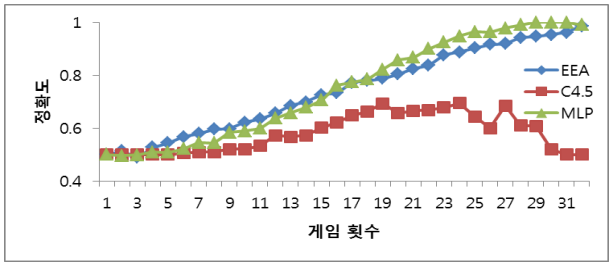

(b) TFT




(c) NTFT




(d) Major




(e) Pav

그림 1. 실험 결과

여기서는 가능한 모든 구성을 테스트 해 볼 수 있기 때문에 여기서 측정한 정확도는 일부 데이터에 과적합 된 것이 아닌 전체 데이터에 대한 정확도이다. 실험 결과에 따르면, AllC를 제외한 모든 플레이어에 대해, 초반 10번 이내의 게임에서는 기존 방법에 비해 더 좋은 결과를 보여줬다.

## 4. 결론

본 논문에서는 게임에서 최소의 데이터 수집으로 특정 상대방에 대한 모델링이 필요한 때, 역공학 알고리즘의 일종인 EEA를 이용할 것을 제안하고, 제안한 방법을 기존 데이터를 이용한 기계학습(C4.5, MLP)와 비교하였다. 실험을 위해 대표적인 IPD의 대표적인 전략 중 다섯 가지를 이용하였다. 실험 결과 알 수 있는 특징은, EEA는 기존 방법에 비해 적은 데이터를 이용해 학습할 때, 상대적으로 높은 성능을 보이는 경향이 있다는 것을 알 수 있었다. 이것은 간접적으로 EEA처럼 기존 경험(데이터)을 이용하여 능동적으로 데이터 수집을 하는 것이 더 유용한 데이터, 즉 과적합을 피할 수 있는 데이터를 수집하는데 도움을 준다는 것을 추론할 수 있다. 하지만, 아직 중요 데이터를 수집한다는 직접적인/통계적인 증명을 위한 추가적인 연구가 필요하다.

## 5. 감사의 글

이 논문은 2013년도 정부(미래창조과학부)의 재원으로 한국 연구재단의 지원을 받아 수행된 중견연구자지원사업(2013-016589) 및 뇌과학 원천기술개발사업임(2010-0018950)

## 참고문헌

[1] B. Weber and M. Mateas, A Data Mining Approach to Strategy Prediction, IEEE Symposium on Computational Intelligence in Games, 2009.

[2] H.-C. Cho, K.-J. Kim and S.-B. Cho, Replay-based Strategy Prediction and Build Order Adaptation for StarCraft AI Bots, IEEE Conference on Computational Intelligence in Games, 2013.

[3] J. Bongard and H. Lipson, Automated Reverse Engineering of Nonlinear dynamical Systems, PNAS, vol. 104, no. 24, pp. 9943-9948, 2007.

[4] G. Kendall, X. Yao and S. Y. Chong, The Iterated Prisoner's Dilemma: 20 Years on, Singapore, Advances in Nature Computation 4, 2007.

[5] H. Park and K.-J. Kim, Opponents Modelling with Incremental Active Learning: A Case Study of Iterative Prisoner's Dilemma, IEEE Conference on Computational Intelligence in Games, 2013.

[6] J. Clune, J.-B. Bouret and H. Lipson, The Evolutionary Origin of Modularity, Proc. of the Royal Society B. 280, 2013.

[7] D. Ashlock and E.-Y. Kim, Fingerprinting: Visualization and Automatic Analysis of Prisoner's Dilemma Strategies, IEEE Transactions on Evolutionary Computation, vol. 12, no. 5, pp. 647-659, 2008.

[8] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, The WEKA Data Mining Software: An Update, SIGKDD Explorations, vol. 11, no. 1, 2009.