JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 25 No. 7, pp. 39-46, July 2020
https://doi.org/10.9708/jksci.2020.25.07.039

Utilization of Simulation and Machine Learning to Analyze and Predict Win Rates of the Characters Battle

Hyun-Syug Kang*

*Professor, Dept. of Computer Science, Gyeongsang National University, Gyeongnam, Korea

[Abstract]

Recently, for designing virtual characters in the battle game field effectively, some methods are very needed to predicate the win rates of the battle of them efficiently. In this paper, we propose a method to solve this problem by combining simulation and machine learning. Firstly, a simulation is used to analyze the win rates of the battle of virtual characters in the battle game. In addition, we apply a regression model based machine learning scheme to predict win rates of the battle of virtual characters according to their abilities. Our experimental results using suggested method show that it is almost no difference between the win rates of the simulation and the prediction results using the machine learning scheme. And also, we can obtain good performance in the experiment using only simple regression based machine learning model.

▸ **Key words**: Character battle, Win rates prediction, Simulation, Regression model, Data analysis

[요 약]

최근, 대전 게임 분야에서, 가상 캐릭터들의 효과적인 설계를 위해 캐릭터의 승률을 효율적으로 예측할 수 있는 방법들이 매우 필요하다. 우리는 본 논문에서 이 문제를 해결하기 위해 시뮬레이션과 기계 학습을 결합하는 방법을 제안한다. 우선 대전 게임에서 가상 캐릭터의 대전 승률을 분석하기 위해서 시뮬레이션을 사용하고, 가상 캐릭터의 능력치에 따라서 승률을 예측하기 위해 회귀 모델에 기반한 기계 학습 기법을 적용한다. 제안한 기법으로 실험한 결과는 시뮬레이션 결과로 나온 승률과 기계 학습 기법이 예측한 승률이 거의 차이가 없다는 것을 확인하였다. 그리고 간단한 회귀 모델에 기반한 기계 학습으로도 실험에서 좋은 성능을 얻을 수 있었다.

▸ **주제어**: 캐릭터 대전, 승률 예측, 시뮬레이션, 회귀 모델, 데이터 분석

• First Author: Hyun-Syug Kang, Corresponding Author: Hyun-Syug Kang
*Hyun-Syug Kang (hskang@gnu.ac.kr), Dept. of Computer Science, Gyeongsang National University
• Received: 2020. 06. 04, Revised: 2020. 07. 13, Accepted: 2020. 07. 14.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

최근, 현실에서 데이터의 수집이 어려운 여러 환경에서 기계 학습의 데이터 생성에 시뮬레이션이 사용되고, 역으로 시뮬레이션 기술에 기계 학습을 위한 에이전트 행동 모델링이 사용되는 등 시뮬레이션 기법과 기계 학습이 서로 상호작용하며 발전하고 있다[1, 2, 3].

본 논문에서는 이의 한 응용으로 시뮬레이션 기법과 기계 학습을 이용하여 대전 게임 분야에서 캐릭터의 설계를 위해 캐릭터의 승률 예측을 효과적으로 수행하는 방법을 연구하였다. 이러한 연구는 대전 게임 분야에서 다양한 캐릭터들을 설계할 때 사전에 그들이 게임에 미치는 영향을 효과적으로 분석하기 위해 매우 필요한 일이다[4, 5].

우선, 우리는 상상의 캐릭터들을 생성하고 대전을 하였을 때 어떤 결과가 나오는지 분석하기 위해 시뮬레이션 기법을 사용하였다. 현실 세계에서 특정 대상의 공격력, 이동 속도 등과 같은 능력치를 바꿔가면서 대전을 하기 에는 물리적, 공간적, 시간적 제약 때문에 대단히 어렵다. 우리는 이를 해결하기 위해 가상 세계에서 캐릭터의 능력치를 설정하고, 캐릭터의 특성에 따라 어떻게 싸우는 것이 좋은지 행동을 상태도(State Diagram)로 모델링한다. 그런 후, 현실의 물리적, 공간적, 그리고 시간적 제약 때문에 불가능한 부분들을 반복해서 실험하는 반복 모의를 통해 능력치를 변화시키면서 대전 결과를 얻었다.

한편, 우리는 반복 모의에 의한 캐릭터 대전의 승률 분석에만 그치지 않고 한발 더 나아가 반복 모의 결과에 대한 승률을 예측하는데 회귀 모델(Regression model) 기반의 기계 학습 기법을 이용하여 예측 효율을 높이는 방법을 설계하였다.

현재까지 우리는 캐릭터 대전의 승률 예측을 위해 시뮬레이션 기법과 기계 학습 기법을 동시에 사용한 연구는 확인하지 못하였다. 우리는 제안 시스템의 구현 결과를 바탕으로 실제 실험 분석을 통해 제안한 방법의 성능을 평가하였다.

논문은 크게 캐릭터 대전 반복 모의를 위한 시뮬레이션 부분과 생성된 데이터를 이용하여 기계 학습 모델을 학습하는 부분으로 구성된다. Ⅱ장에서 가상 캐릭터 대전을 위한 캐릭터 모델링과 반복 모의를 설명하고, 이를 이용해 생성한 캐릭터들의 대전 승률을 얻는다. Ⅲ장에서는 반복 모의 결과를 활용하여 캐릭터의 대전 승률을 예측하는 기계 학습 모델을 개발한 후, 기계 학습 모델이 예측한 승률 결과가 잘 학습되었는지 평가한다. 마지막 Ⅳ장에서는 결론을 기술하고, 향후 연구 계획에 대해 언급한다.

II. Simulation for Virtual Character Battle

이 장에서는 시뮬레이션 기법을 활용하여 가상 캐릭터 대전을 모델링하고, 반복 모의를 통해 대전 결과를 분석한다. 일반적으로 대전 결과를 분석하기 위해서는 시뮬레이션 모델 중에서 공학 혹은 교전급 모델이 필요하며, 실시간 모의가 가능해야 한다[6]. 이를 위해 우리는 산업 공학 등에서 활발하게 사용되고 있는 애니로직(AnyLogic)이라는 시뮬레이션 도구를 사용하였다. 이때 에이전트 기반 모델링(ABM, Agent-Based Modeling)[7] 기법으로 가상 캐릭터의 능력치와 행동(Action)을 정의하여 캐릭터를 모델링하여 대전 결과를 얻었다.

1. Agent-based Modeling Using AnyLogic

애니로직은 AnyLogic(구: XJ Technologies)사에서 개발한 복합 모델링 도구이다[8]. 애니로직은 [Fig. 1]과 같이 직관적인 사용자 인터페이스(User Interface)를 제공하기 때문에 전문 지식이 없는 사람도 간단한 순서도 형태로 알고리즘의 구현이 가능하다. 애니로직에서 에이전트 기반 모델링은 표현하고 싶은 부분을 하나의 캐릭터, 즉 에이전트(Agent)로 표현하여 특성(Feature)과 행동(Action)을 정의하고 모델링한다. 일반적으로는 여러 개의 에이전트를 생성하여 각각의 에이전트 개체가 자신이 가진 목표를 달성하기 위해 행동을 선택하도록 하며, 이때 생성되는 결과를 분석할 수 있다.

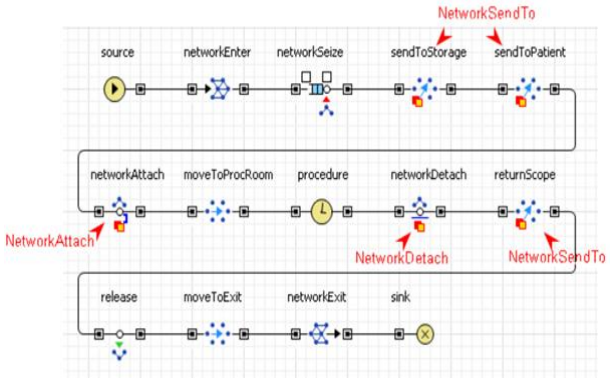

Fig. 1. Example of AnyLogic Simulation

[Fig. 2]는 에이전트 기반 모델링의 예를 그림으로 표현한 것이다. 실제 상상하는 캐릭터는 굉장히 정밀도가 있어, 이를 모두 가상의 세계에 표현하기에는 많은 어려움이 있다. 따라서 가상의 캐릭터에서 꼭 필요한 부분을 식별하고, 행동을 정의해야 한다. 그러면 최종적으로 특성과 목표를 위한 행동이 반영된 가상의 에이전트가 생성된다. 여

기서 에이전트의 특성은 어떤 숫자 값이나 참, 거짓과 같은 입력 정보로 표현할 수 있다.

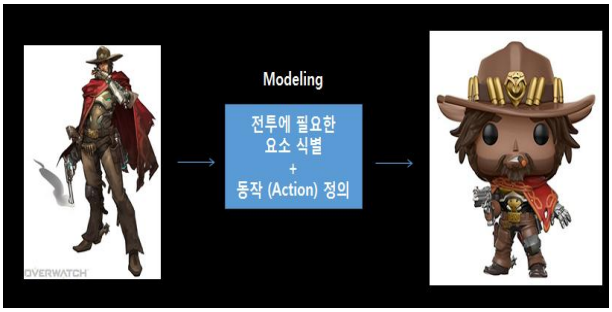

Fig. 2. Example of Agent-based Modeling

[Fig. 3]은 에이전트의 특성과 행동을 정의하여 모델링한 예이다. 특성은 속도, 체력 등과 같은 정수 형태의 값으로 표현할 수 있다. 이 값들은 캐릭터의 능력치가 되는데 이런 능력치를 활용하여 상태도에서 적을 이기기 위해 어떤 행동을 할지 결정한다. 행동을 표현하는 상태도에는 각각에 대한 상태(State)를 정의하게 되는데, 특정 조건에 따라서 다음 상태로 바뀌게 되고, 더 이상 상태를 변경할 필요가 없을 경우에는 그 상태가 종료 상태에 머물게 된다.

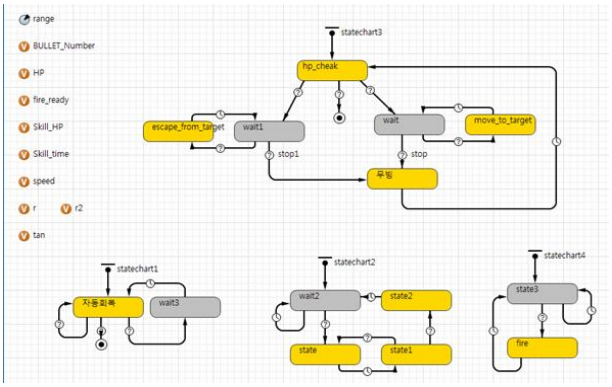

Fig. 3. Design Characters Using Agent-based Modeling

2. Virtual Character Modeling

실험을 위해 본 논문에서는 2개의 가상 캐릭터의 특성을 표현하고, 적을 이기기 위한 행동을 정의한다. 우선, 캐릭터 맥크리와 캐릭터 리퍼는 아래 [Fig. 4]와 같은 능력치와 행동으로 구성하고, 이는 애니로직에서 에이전트로 구현하였다.

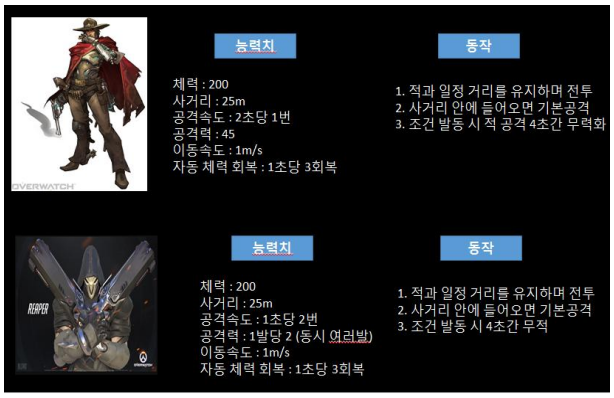

Fig. 4. Two Virtual Characters: Mackly and Reaper

두 캐릭터의 가장 큰 모델링 차이는 탄환 모델링에 있는데, 캐릭터 맥크리는 적에게 탄을 단발로 쏘는데 반해 캐릭터 리퍼는 동시에 탄을 여러 발 발사하며, 대신 각 탄의 공격력은 낮게 하였다. ([Fig. 5] 참조)

Fig. 5. Single Bullet(Left) and Multiple Bullets(Right)

이렇게 캐릭터 설계의 차이 때문에 승리를 위한 능력치는 어떤 캐릭터냐에 따라 다를 것인데, 이는 실제 시뮬레이션을 이용한 반복 모의를 통해 승리 여부를 분석해야 한다. 그 외 체력, 사거리, 공격 속도와 같은 능력치가 변화할 때의 승리 여부를 파악하기 위해서는 캐릭터 설계와 별개로 입력 값들을 설정하는 모델링 기법이 추가로 필요하다.

[Fig. 6]은 맥크리와 리퍼 캐릭터들의 능력치를 애니로직에 구현한 화면이다. 캐릭터를 위한 능력치들은 왼쪽과 같이 하나의 값으로 표현하였다. 그중 tan의 값을 보면 수학 모델링을 위한 함수 Math.tan를 사용하여 값을 정하고, 이 값은 행동에 따라 계속해서 변하게 하였다.

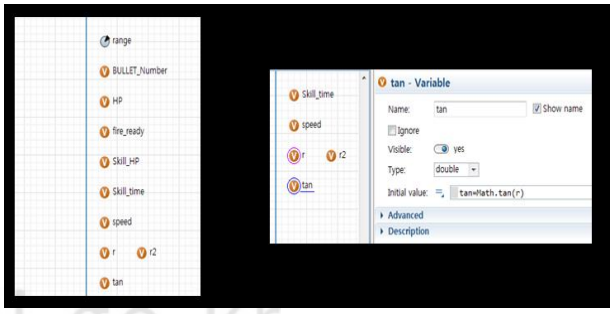

Fig. 6. Capability and Status Modeling

[Fig. 7]과 같이 행동은 왼쪽의 상태도로 표현하고, 그 안의 세부 내용은 오른쪽의 간단한 자바(Java) 코드로 구현 가능하다. 오른쪽의 코드는 이동 상태의 세부 구현이고, 여기서는 r 값을 계산하고, 다시 x, y 좌표를 갱신하여 캐릭터가 그 좌표로 이동하도록 정의하였다. 해당 상태는 입구 행동(Entry action)에 상세 내용이 있는데, 필요에 따라 그 상태를 벗어나 다른 상태로 변할 때는 해당 내용을 출구 행동(Exit action)에 표현할 수도 있다.

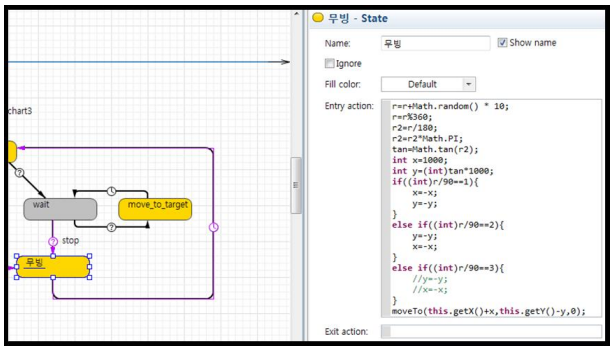

Fig. 7. Character Moving State

캐릭터 외형을 위한 3D 모델링은 애니로직에서 기본으로 제공하는 3D 모델을 가공하여 사용하였다. 여기서 캐릭터가 가상공간에서 어느 정도의 크기로 표현될지 결정하는 척도가 있는데, 이 크기에 따라 캐릭터가 탄에 맞을 수도 있고, 그렇지 않을 수도 있기 때문에 적절한 캐릭터 크기를 설정해야 한다. 여기서는 [Fig. 8]과 같이 캐릭터 척도를 설정하고, 대전 모의실험을 수행하였다.

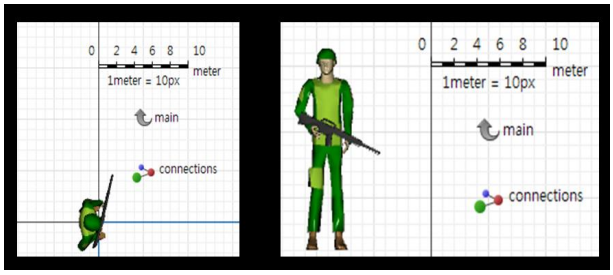

Fig. 8. Design of Character Reaper

3. Capability Modeling and Win Rates Analysis

능력치를 변화시켰을 때, 가상 캐릭터의 대전 승률을 얻기 위해서 하나의 능력치에 대해서 100회의 반복 모의를 수행한 다음, 이를 산술 평균하여 승률을 얻었다. 반복 모의를 수행할 때, 변하는 능력치 외의 나머지 능력치들은 모두 고정하여 똑같은 환경에서 모의가 이루어지도록 하였다. [Fig. 9]는 전투 화면의 예이다.

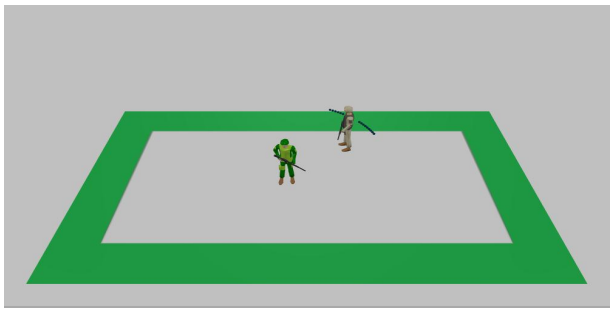

Fig. 9. Battle of Virtual Characters Using AnyLogic

반복 모의를 수행할 때 효과 척도로 사용할 능력치는 이동 속도, 공격력, 공격 속도로 정하였고, 그 때의 능력치들은 [Table 1]과 같이 설정하였다. 이동 속도에 따른 승률 분석을 100번 수행하였을 때, 이동 속도 10m/s에서 리퍼가 맥크리와 대전하여 약 56% 승리를 하였다. 이동 속도를 20m/s로 변경하였을 때, 리퍼는 약 91%의 높은 승률을 보였다.

Table 1. Input Capabilities of Charater Reaper for Experiment

Ability	Moving Speed (m/s)	Attack Power	Attack Speed (Number/s)
Input Value	2 ~ 20	1 ~ 10	0.1 ~ 1

[Fig. 10]은 능력치 중에서 이동 속도를 변화하였을 때, 리퍼의 승률을 나타낸 그래프이다. 이동 속도가 10m/s일 경우, 리퍼의 대전 승률이 약 56%로 높게 나타났다. 이동 속도가 20m/s일 경우, 리퍼의 대전 승률은 약 97%이며, 대부분의 경우 승리하는 것을 알 수 있다.

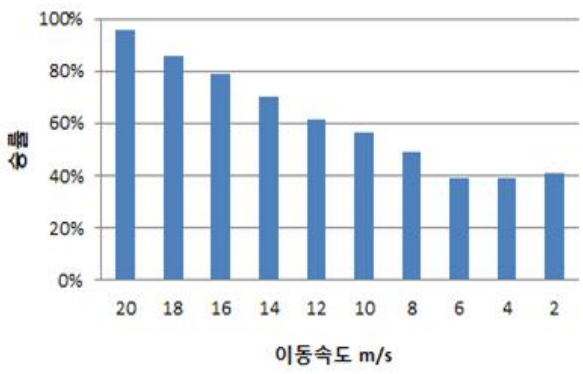

Fig. 10. Win Rates of Reaper According to Moving Speed

[Fig. 11]은 능력치 중에서 공격 속도를 변화하였을 때, 리퍼의 승률을 나타낸 것이다. 공격 속도는 초당 공격 횟

수로 나타나는데, 예를 들어 공격 속도 1은 캐릭터가 초당 1번 공격하는 것을 의미한다. 공격 속도가 0.5일 경우, 리퍼는 승률이 약 58%로 나타나며, 0.3 보다 더 빠르게 공격할 경우는 거의 승률이 100%에 수렴하였다.

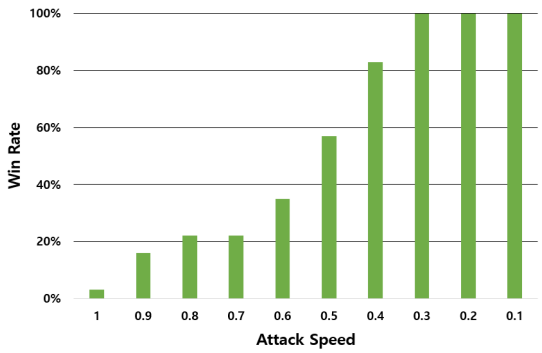

Fig. 11. Win Rates of Reaper According to Attack Speed

리퍼 캐릭터의 대전 승률 분석을 위해 이동 속도와 공격 속도를 효과 척도로 설정한 것은 승률이 변하는 것을 확인할 수 있으므로 타당하다고 볼 수 있다. 반복 모의 시 이동 속도에 대해서는 8m/s에서 승률이 50%를 넘어서고, 그 이후로는 승률이 계속해서 증가하므로 민감도 분석 측면에서 이동 속도는 8m/s 이상이 필요한 것으로 분석된다. 공격 속도 측면에서는 공격 속도가 0.5 이상일 경우 승률이 50%를 넘기 때문에 공격 속도 0.5를 기준 으로 캐릭터의 능력치를 설계해야 함을 알 수 있다.

4. Data Creation for Simulation of Virtual Characters

Ⅱ.3 절에서 설명한 내용을 응용하여, 시뮬레이션으로 두 가상 캐릭터의 능력치인 이동 속도, 공격력, 공격 속도를 변화하였을 때 승패 결과 데이터를 생성하였다. 반복 모의를 통해 얻은 결과는 각 능력치 및 상태와 함께 저장하였고, 총 5천개의 데이터를 생성하였다.

III. Predict Win Rates of Character Battle Using Machine Learning

앞 장의 시뮬레이션 기반 대전 캐릭터 승률 분석 기법은 다양한 능력치에 따라 승률을 구할 수는 있지만, 반복 모의에는 많은 시간과 자원이 필요하다. 따라서 기존 결과를 가지고 있지 않을 경우에 매번 계속해서 반복 모의를 하는 것은 현실적인 문제를 해결하는데 매우 제한적이다. 이 경우, 기계 학습을 활용하여 수많은 반복 모의 결과를 학습하면 해당 문제를 효과적으로 해결할 수 있다. 즉, 반복 모의를 수행할 때, 입력 값이 제대로 들어갔는지, 결과는 올바른지 등을 검증하는 시간을 매우 많이 줄일 수 있다. 추가로 다른 능력치에 대한 수정이 일어나면 해당 특징을 입력에 활용함으로써 기계 학습 모델을 새로 구현할 필요없이 그대로 재학습만 하면 된다. 이 장에서는 이 방법을 제안한다.

1. Machine Learning Model

1) Linear Regression Model

통계학에서, 선형 회귀(線型回歸, Linear Regression)는 종속 변수 y와 한 개 이상의 독립 변수(또는 설명 변수) x와의 선형 상관관계를 모델링하는 분석 기법이다. 한 개의 설명 변수에 기반하는 경우에는 단순 선형 회귀, 둘 이상의 설명 변수에 기반 하는 경우에는 다중 선형 회귀라고 한다[9].

이 선형 회귀 모델을 기반으로 하는 기계 학습에서는 인공신경망(Artificial Neural Network)을 이용하여 모델을 구현하고, 이를 학습한다. 이때 우리가 사용한 선형 회귀 모델은 다음 식 (1)의 간단한 선형 방정식 형태로 표현할 수 있다.

$$y = w * x + b \quad \text{.....} \quad (1)$$

여기서 x는 학습하고자 하는 입력 데이터가 되며, 일반적으로 학습하고자 하는 특성을 표현하는 것을 선정한다. 기계 학습에서 w는 가중치를 나타내는데, 입력 x와 조합하여 결과 y'를 구한다. 이 결과 y'는 선형 회귀 모델에서 계산된 예측 값인데, 이 값을 실제 결과 y와 비교하여 모델의 정확도를 검사하게 된다.

2) Logistic Regression Model

로지스틱 회귀(Logistic Regression)는 독립 변수의 선형 결합을 이용하여 사건의 발생 가능성을 예측하는데 사용되는 통계 기법이다[10]. 로지스틱 회귀의 목적은 일반적인 회귀 분석의 목표와 동일하게 종속 변수와 독립 변수 간의 관계를 구체적인 함수로 나타내어 향후 예측 모델에 사용하는 것이다. 이는 독립 변수의 선형 결합으로 종속 변수를 설명한다는 관점에서는 선형 회귀와 유사하다. 하지만 로지스틱 회귀는 선형 회귀 분석과는 다르게 종속 변수가 범주형 데이터를 대상으로 하며 입력 데이터가 주어졌을 때 해당 데이터의 결과가 특정 분류로 나뉘기 때문에 일종의 분류(Classification) 기법으로도 볼 수 있다.

2. Experimental Environment of Machine Learning

구글 코랍(Google Colab)은 구글에서 제공하는 데이터 분석이 가능한 실험 환경으로 별도의 설치가 필요 없으며 완전히 클라우드에서 실행되는 무료 주피터(Jupyter) 노트 환경이다[11].

[Fig. 12]는 구글 코랍의 주피터 개발 환경 화면인데, 파이썬 코드를 웹에서 실행하고 그 결과를 바로 볼 수 있다. 추가적으로, 구글의 딥러닝(Deep Learning) 라이브러리인 텐서플로우(Tensorflow)를 기본으로 제공하며, 그 외 다른 딥러닝 라이브러리(예, Pytorch) 등도 설치하여 사용 가능하다[12].

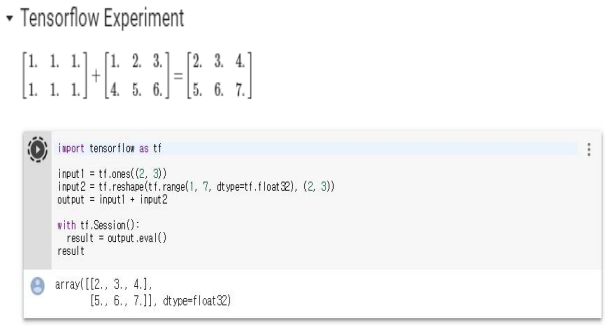

Fig. 12. Implementation Screen of Google Colab

구글 코랍은 무료이면서 강력한 CPU와 GPU 자원을 사용하도록 지원하고, 각종 구글 서비스와 연동이 쉽기 때문에 본 논문에서는 기계 학습 모델 실험 환경으로 이를 사용한다. [Table 2]는 우리가 사용한 구글 코랍 개발 실험 환경의 사양이다.

Table 2. Experimental Environment of Google Colab Development

	Colab	PC
CPU	Intel® Xeon® CPU @2.30Hz	Intel Core i5 8500
RAM	13GB	8GB
GPU	Tesla K80	GeForce GTX 1060 3GB
Storage	33GB	579GB
S/W	Pytorch	Window 10

3. Design of Machine Learning Model for Predicting Win Rates of Characters

Ⅱ.4절에서 반복 모의로 생성한 입력 데이터를 활용하여, 여기서는 가상 캐릭터의 대전 승률을 예측하는 기계 학습 모델을 설계한다. 승률 예측 문제는 회귀 모델로 해결이 가능한 문제이며, 기본적인 선형 모델 대신 딥러닝에서 사용하는 여러 가지 기능들을 추가하였다. 대표적으로 딥러닝에서 선형 모델의 성능을 향상시키기 위해 인공신경망에 비선형성을 부여하는 활성 함수(Activation Function)인 ReLU가 사용되었다[13]. 아래 [Fig. 13]은 딥러닝 라이브러리인 파이토치(Pytorch)를 사용하여 구현한 기계 학습 모델이다.

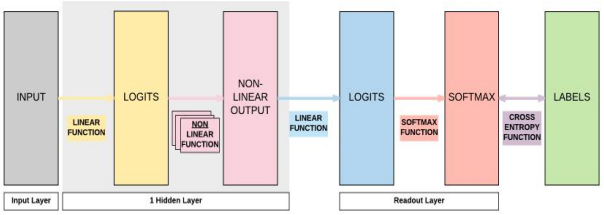

Fig. 13. General Machine Learning Model for Predicting of Win Rates

구글 코랍에서 해당 부분을 구현하였을 때, 핵심적인 파이토치 코드는 [Fig. 14]와 같다. 여기에 작성한 코드에서 입력 데이터를 모의실험에서 생성된 능력치들로 설정하면 대전 승률에 대한 학습을 할 수 있다.

```
model  = nn.Linear(1, 1, bias=True)
model.weight, model.bias
cost_func = nn.MSELoss()
optimizer = torch.optim.SGD(model.parameters(), lr=0.01)
model(x)
plt.ion()
for step in range(10000):
    prediction = model(x)
    cost = cost_func(prediction, y)
    optimizer.zero_grad()
    cost.backward()
    optimizer.step()
plt.scatter(x.data.numpy(), y.data.numpy())
plt.plot(x.data.numpy(), prediction.data.numpy(), 'b--')
plt.title('cost=%.4f, w=%.4f, b=%.4f' % (cost.data[0],model.weight
plt.show()
plt.ioff()
```

Fig. 14. Pytorch Code for Predicting Win Rates

본 논문에서는 [Fig. 13]의 기계 학습 모델 구조를 바탕으로 [Fig. 14]와 같이 파이토치 코드 형태로 기계 학습 모델을 구현하였다. 기계 학습 모델은 경사하강법(SGD, Stochastic Gradient Decent) 알고리즘을 사용하였고, 학습률(Learning rate)은 0.01을 고정으로 사용하였다. 학습을 위한 특성(Feature)은 이동 속도, 공격력, 공격 속도로 정하였고 입력값은 Ⅱ.4절에서 생성한 데이터를 사용하였다. 우리는 [Fig. 15]와 같이 기본적인 선형 모델(모델 1)과 ReLU 등과 같이 딥러닝에서 비선형성을 부여한 모델(모델 2)을 설계하였고, 두 모델의 성능을 비교 분석하였다. 은닉층(Hidden Layer)는 모두 동일하게 10개를 사용하였다.

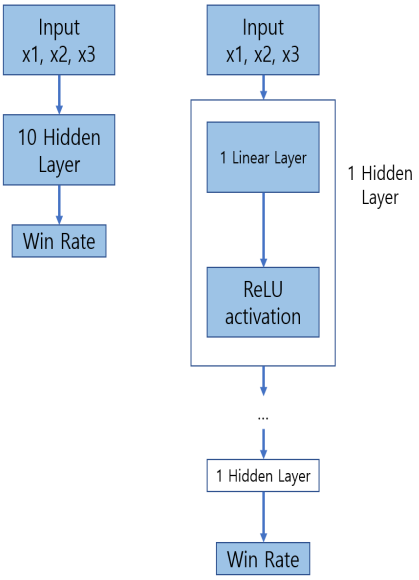

Fig. 15. Design of Machine Learning Model for Predicting Win Rates

4. Performance Evaluation of Machin Learning Model

본 논문에서는 앞에서 언급한 능력치별 가상 캐릭터의 승률 데이터 5천 건에 대해서 [Fig. 16]과 같이 학습 집합(Train Set)과 검사 집합(Test Set)으로 구분하였다[14]. 입력 특징인 이동 속도, 공격력, 공격 속도에 대해서 전체 데이터 중 60%인 3천 건을 기계 학습 모델의 학습에 사용하였다. 나머지 40%인 2천 건에 대해서는 학습된 기계 학습 모델을 평가하기 위한 검사용으로 사용하였는데, 우리는 2천 건에 대해서 승률 예측을 얼마나 잘하는지에 대해 성능을 측정하였다.

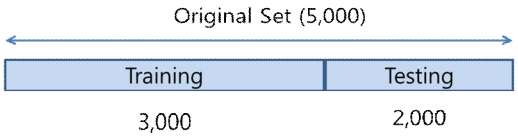

Fig. 16. Setting of Input Data for Evaluation

구체적으로 학습 집합은 공격력, 공격 속도, 이동 속도에 대한 입력 특성을 가지며, 3천 건에 대해서 반복해서 학습을 진행하였다. 즉, 학습에 사용된 능력치들은 수식 (1)에서 x에 해당하며, x는 세부적으로 $x1$, $x2$, $x3$로 나누어진다고 할 수 있다. 모델 성능 평가를 위해서 학습에 사용되지 않은 2천 건의 입력 데이터를 검사 집합으로 사용하였고, 모델의 결과로 예측된 값과 모의실험의 결과로 나온 실제 승률 결과를 비교하였다. 기계 학습 모델 1, 모델 2는 [Fig. 17]과 같이 3천 번을 반복하여 실험하였다.

```
Iteration: 500. Loss: 0.3877025246620178. Accuracy: 90
Iteration: 1000. Loss: 0.1337055265903473. Accuracy: 93
Iteration: 1500. Loss: 0.2038637101650238. Accuracy: 95
Iteration: 2000. Loss: 0.17892278730869293. Accuracy: 95
Iteration: 2500. Loss: 0.14455552399158478. Accuracy: 96
Iteration: 3000. Loss: 0.024540524929761887. Accuracy: 96
```

Fig. 17. Repeated Experiment of Model 1 and Model 2

반복해서 실험한 결과, [Table 3]과 같이 기계 학습 모델 1에 대해서는 0.87의 정확도를 얻었고, 모델 2에 대해서는 0.96의 정확도를 얻었다. 딥러닝에서 비선형성을 부여하여 기존의 인공신경망의 성능을 급격하게 향상시킨 부분을 고려하였을 때[13], 캐릭터 대전 승률 데이터에도 비선형성을 부여한 기계 학습 모델이 더 적합하다는 사실을 알 수 있다. 은닉 층의 수를 늘리거나 최근 주목받고 있는 CNN 구조를 적용한다면 보다 정확도를 향상시킬 수 있을 것으로 기대한다[14].

Table 3. Predicting Win Rates Between Model 1 and Model 2

Machine Learning Model	Accuracy
Model 1 - Linear	0.87
Model 2 - Non-Linear	0.96

IV. Conclusions

현실에서 실험하지 못하는 대전 게임의 승률 분석을 위해 캐릭터를 가상의 세계에서 분석하는 시뮬레이션 기법을 사용하고, 그 결과로 나오는 데이터를 활용하여 기계 학습으로 승률을 예측하는 작업을 수행하였다. 시뮬레이션은 현실 세계를 가상 세계로 옮기고, 반복 실험을 통해 특정 상황에 대해 경향을 알 수 있게 한다는 장점이 있다. 하지만 반복 모의를 수행할 때는 많은 시간과 노력, 실험 조건을 구성해야 하는데, 이는 시뮬레이션의 활용을 저하시키는 큰 요인이 된다. 또한, 반복 모의 결과로 생성된 데이터를 해석하는 데는 많은 통계적인 지식이 요구되며, 입력 데이터와 행동이 적합한지 검증하는 단계가 필수적이므로 분석에 많은 어려움이 있다. 이런 상황에서 기계 학습을 활용하여 기존에 모의실험으로 생성된 데이터를 학습할 경우, 차후 능력치 변화에 대한 승률 예측을 통해 적은 시간에 결과를 예측할 수 있다.

우리는 여기서 처음으로 시뮬레이션 기법과 기계 학습

을 같이 사용하면 캐릭터 대전의 승률 예측을 보다 효율적으로 수행할 수 있다는 것을 확인하였다. 우리는 기계 학습의 간단한 모델인 로지스틱 회귀 모델을 활용하였는데, 예측 성능을 올리기 위해 모델 최적화를 수행한다면 더 좋은 성능을 얻을 수 있으리라 기대한다.

향후, 다른 입력 값을 반영하고 모델의 최적화를 통해 기계 학습에 요구되는 자원들을 대폭 낮추는 연구를 진행할 예정이다. 우리가 제안한 시뮬레이션과 기계 학습을 조합한다면, 현실의 다른 많은 문제들에 대해서도 미래에 어떤 결과가 나타날지 효과적으로 미리 경향을 분석할 수 있을 것이다.

ACKNOWLEDGEMENT

Author appreciate Dr. Yeong-Ung Seo and students of 2018 Study Under Teacher Course at Gyeongsang National University Science Gifted Education Center for their helps to prepare this paper.

REFERENCES

[1] Schwab. K., “The Fourth Industrial Revolution: What It Means, How To Respond,” World Economic Forum, 14 Jan 2016. www.weforum.org.

[2] Department of Defense(USA), "Modeling and Simulation(M&S) Verification, Validation, and Accreditation(VV&A)," DoD Instruction 5000.61, Oct 15, 2018.

[3] The Korea Society for Simulation, “Topic: Simulation and Artificial intelligence”, Dec 2018.

[4] Roohi, S., Takatalo, J., Guckelsberger, C., and Hämäläinen, P., “Review of Intrinsic Motivation in Simulation-based Game Testing.” In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, pp. 1-13. Apr 2018. DOI: https://dl.acm.org/doi/10.1145/3173574.3173921

[5] Guerrero-Romero, C., Lucas, S. M., and Perez-Liebana, D., “Using a Team of General AI Algorithms to Assist Game Design and Testing.” In 2018 IEEE Conference on Computational Intelligence and Games, pp. 1-8, Aug 2018. DOI:https://www.researchgate.net/publication/328308938

[6] Tag-Gon Kim, “M & S Engineering,” The Korean Institute of Information Scientists and Engineers, Vol. 25, No. 11, pp. 5-15, Nov 2007. DOI:https://www.koreascience.or.kr/article/JAKO200735822312725.page

[7] Si-Heon Kim, “Agent-Based Modeling and Simulation Methodology using Social-Level Characteristics: A Case Study on Self-Adaptive Smart Grid and Military Domain Systems using Tropos,” Journal of KIISE, Vol. 42, No. 12, pp. 1503-1521, Dec 2015. DOI:https://www.koreascience.or.kr/article/JAKO201502648775165.page

[8] The AnyLogic Company, “What is AnyLogic?”, 2018. www.anylogic.com,

[9] Freedman. D. A., “Statistical Models: Theory and Practice”, Cambridge University Press, 26, 2009.

[10] Cox. D. R., “The Regression Analysis of Binary Sequences,” Journal of the Royal Statistical Society, Vol. 20, No. 2, pp. 215-242, 1958. DOI:https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1958.tb00292.x

[11] Google Colab, “Hello, Colaboratory”, 2018. https://colab.research.google.com.

[12] The Pytorch Company, “From Research To Production”, 2018. https://pytorch.org/.

[13] Krizhevsky. A,, Sutskever. I,, and Hinton. G. E., "Imagenet Classification with Deep Convolutional Neural Networks," In Advances in Neural Information Processing Systems(NIPS), pp. 1106-1114, 2012. DOI:http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networ

[14] Seong-hun Kim, “A Machine Learning For Everyone – Deep Learning Lectures”, 2016. https://hunkim.github.io/ml/.

Authors

Hyun-Syug Kang received the M.S. and Ph.D. degrees in Computer Science and Statistics from Seoul National University in 1983 and 1989, respectively. During 1981-1985, he stayed in ETRI. And also during 1985-1993, he stayed in Chonbuk National University. He has been a professor at Gyeongsang National University since 1994. His research interests are multimedia, database, and intelligent system.