Deep Learning 기반 기계학습 알고리즘을 이용한 야구 경기 Big Data 분석

Big Data Analysis based on Deep Learning for Baseball Game Data

저자 (Authors)
김종훈, 김경태, 한종기
Jong-Hun Kim, Kyung-Tae Kim, Jong-Ki Han

출처 (Source)
한국통신학회 학술대회논문집 , 2015.11, 262-265(4 pages)
Proceedings of Symposium of the Korean Institute of communications and Information Sciences , 2015.11, 262-265(4 pages)

발행처 (Publisher)
한국통신학회
Korea Institute Of Communication Sciences

URL
http://www.dbpia.co.kr/journal/articleDetail?nodeId=NODE06564907

APA Style
김종훈, 김경태, 한종기 (2015). Deep Learning 기반 기계학습 알고리즘을 이용한 야구 경기 Big Data 분석. 한국통신학회 학술대회논문집, 262-265

Deep Learning 기반 기계학습 알고리즘을 이용한 야구 경기 Big Data 분석

김종훈, 김경태, 한종기
세종대학교

Jonghun1277@gmail.com, kimkt6157@naver.com, hjk@sejong.edu

Big Data Analysis based on Deep Learning for Baseball Game Data

Jong-Hun Kim, Kyung-Tae Kim, Jong-Ki Han
Sejong University

요 약

최근 Big Data 분석을 위한 연구가 중요하게 여겨지면서, 이에 대한 연구와 관련 기술 개발이 다양한 형태로 이루어지고 있다. 본 논문에서는 Big Data 분석을 위해 Deep Learning 기법을 이용하는 기계학습 알고리즘을 개발하여 일상 생활에서 흔하게 접하는 Big Data 에 적용한 실험 결과를 제시한다. 구체적으로는 우리 나라 국민들에게 인기를 끌고 있는 프로야구 경기에서 발생한 지난 30 년간의 빅데이터를 분석하여, 2015 년 시즌의 KBO 야구경기 승패를 예측하는 알고리즘을 제안한다.

Ⅰ. 서론 [1]

최근 빅데이터 신호처리 기술의 중요성이 대두되면서 그에 대한 연구와 개발이 다양한 형태로 진행되고 있다. 이를 위해서 연구되고 있는 많은 기술들 중에 대표적인 기술로는 Neural Network 에 기반하여 설계된 Deep Learning 기술이 있다.

Deep Learning 은 컴퓨터 인공지능 학습법 중 하나이며 인간의 사고와 판단 과정을 모방한 인공지능(AI) 기술로 사람의 사고방식을 컴퓨터에 가르치는 기계학습의 한 분야이다.

본 연구는 2015 시즌 야구 경기의 결과 예측을 위하여 지난 1982 년도부터 야구 경기에서 발생한 수많은 데이터를 사용하였다. 33 년간의 데이터를 가공하고 이를 Deep Learning 기법으로 분석해 보았다. Deep Learning 을 통한 분류 및 예측에 관한 많은 연구가 이루어지고 있다.

이에 본 연구에서는 지난 과거의 야구 경기의 데이터를 통해 2015 시즌 야구 경기의 예측을 위해 Deep Learning 을 도입하였다. Deep Learning 을 통해 나온 실험 결과 값과 실제 경기의 승률과 비슷하게 예측하는 것에 목적을 두고 있다.

본 논문에서는 Neural Network 를 사용하여 현재 가지고 있는 데이터 값으로 미래에 있을 경기의 승패를 맞추는 모델을 구현하였다. 과거부터 2015 년 9 월 5 일까지의 데이터를 사용하여 2015 년 9 월 30 일까지의 승률을 예측해보았다.

[1] 연락저자: 한종기

이 논문은 2015 년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-2015R1A2A2A01006193).

Ⅱ. 제안하는 Deep Learning 기술

A. 야구 경기의 Big Data 특성

K-ICT 빅데이터 센터와 sports2i, KBO 공식 홈페이지에서 얻은 1982 년부터 2015 년 9 월 5 일까지의 선수 개인별, 팀 별 데이터를 활용하였다

B. 제안하는 Neural Network 모델

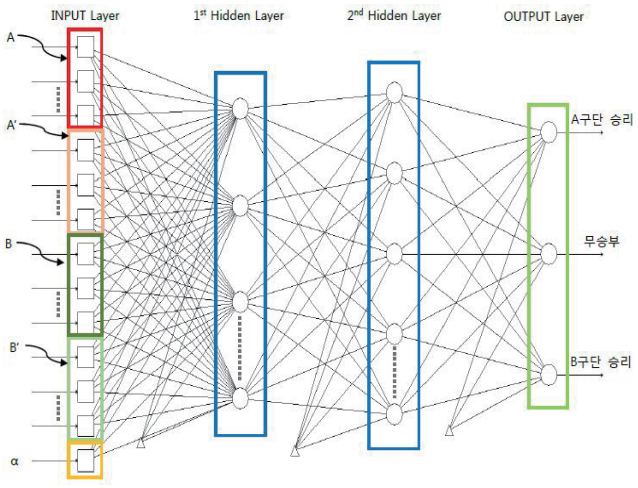

Figure 1. Neural Network 모델

Neural Network 는 인간의 뇌 기능을 모방하려는 생각을 기초로 두고 있다. 인간에게는 아주 간단하고 당연한 사고방식을 컴퓨터에 학습시키려는 것이다. 연결선의 결합으로 네트워크를 형성한 Node 가 학습을 통해 연결선의 결합 가중치(weight)를 변화시켜 학습하고 문제를 해결할 수 있는 알고리즘이다. [5]

대부분의 Neural Network 는 다수의 Node 로 구성된 몇 개의 Layer 로 구성 되어 있으며, 각 Node 는 외부 또는 다른 Node 로부터 다수의 입력 값을 받으며, 각 Node 는 입력 값과 입력이 들어오는 연결선의 가중치(w, v)를 특정한 수학적 함수를 통하여 변환하여 출력 값으로 산정하게 되며, 이를 다른 Node 의 입력 값 또는 Output 으로 사용하게 된다. 연결선에 할당된 가중치들은 일정한 수학적 알고리즘을 사용하여 변한다.

Neural Network 에서 에러를 줄이기 위해 가중치(weight)를 학습한다. Weight 를 최적화하는 방법으로 Feed-forward Propagation 을 통해 나온 Output 값과 Target 값으로 Output Layer 부터 Input Layer 까지 순서대로 Back Propagation 을 사용하여 오차 수정하여 최적화 한다. [2]

Table 1. Neural Network 의 구조

	Input Layer	Hidden Layer 1	Hidden Layer 2	Output Layer
Node 개수	33	8	9	3
Transfer function	·	Log-Sigmoid	Log-Sigmoid	Tangent-Sigmoid

Transfer Function, Layer 의 개수, Node 의 개수를 여러 가지 방법으로 조합하여 그 중 최고의 성능을 내는 Neural Network 를 만들었다. 그 결과 Layer 의 개수는 3 단이며 첫 번째 Hidden layer 에서 Node 의 개수는 8 개로 설정하고 두 번째 Hidden layer 에서는 9 개로 설정하다. Transfer Function 의 종류는 Hidden Layer 에서 Log-Sigmoid 함수를 사용하고 Output layer 에서는 Tangent-Sigmoid 함수를 사용하는 것이 최고의 학습상태를 보였다.

C. Deep Learning 의 입력 및 출력 정보

Figure 1 과 같이 야구경기 승패를 맞추기 위한 Neural Network 의 구조는 33 개의 Input(A, A', B, B')과 3 개의 Output(A 팀 승리, B 팀 승리, 무승부), 2 단의 Hidden Layer(1st Layer 의 Neuron 개수: 8 개, 2nd Layer 의 Neuron 개수: 9 개)로 설정하였다. Hidden Layer 에서 Transfer Function 은 Log-Sigmoid 함수를 사용하였으며, Output Layer 에서는 Tangent-Sigmoid 를 사용하였다, Output Layer 에서는 Tangent-Sigmoid 를 사용하였다

Table 2. Input 종류

INPUT 종류	Input Node 종류
A	평균자책점, 타율, 피안타수, 볼넷, 투구이닝, 삼진, 탈삼진, 승률, 홈 or 어웨이
A'	평균자책점, 타율, 피안타수, 투구이닝, 삼진, 탈삼진, 승률
B	평균자책점, 타율, 피안타수, 볼넷, 투구이닝, 삼진, 탈삼진, 승률, 홈 or 어웨이
B'	평균자책점, 타율, 피안타수, 투구이닝, 삼진, 탈삼진, 승률

33 년간의 시즌 별(연 단위)로 데이터를 입력시킴으로써 각 구단들의 지난 성적과 추세를 파악할 수 있다. 이러한 완만한 데이터로 Neural Network 의 weight 를 학습시켜 2015 년도 시즌 데이터(하루 단위)를 사용할 때 Error 가 보다 적게 나올 수 있게 초기화시켰다.

Deep Learning 으로 보다 인간다운 사고방식으로 예측시키기 위해 (A 구단, A 구단')와 (B 구단, B 구단')으로 세분화 하여 예측하였다. A, B 에서는 현재의 값을 생각하여 Input 값을 넣었지만 사람이 생각할 때 과거의 추세를 보고 미래를 예측을 하듯이 A', B' 에서 지난 시즌과 최근 3 경기의 성적을 넣어 모델을 개선시켰다. 각각 A', B' 를 추가하여 입력시킴으로써 구단 별 성적의 상승세와 하강세 흐름을 통해 경기의 승패를 예측할 수 있게 설계했다.

Neural Network 를 통한 야구 경기 예측 모델을 만들면서, 과연 입력 Node 를 어떠한 stat 들로 구성하여야 보다 정확한 예측을 하기 위해 결정하였다. 우선 투수 측면에서 중요하다고 판단하여 Input Node 로 설정한 stat 은 평균자책점과 피안타수, 볼넷, 투구이닝, 탈삼진, 승률이다. 타자 측면에서 중요 stat 으로 뽑은 것은 타율과 삼진이다. 상대적으로 타자의 stat 이 적은 이유는 야구경기에서 득점을 내는 타자보다 실점을 적게 하는 투수가 경기의 승패에 보다 중요하기 때문이다. 그리고 경기 외적인 요소로 작용하는 Home 구장(1) / Away 구장(0)도 stat 중 하나로 설정하였다.

Table 3. Target 의 종류

TARGET	1982~2014 년 시즌 별 데이터	2015 년도 시즌 일일 데이터
A 구단 승	$0 \le \text{Output} \le 1$	$Output = \begin{cases} 0 & A\text{구단 패} \\ 1 & A\text{구단 승} \end{cases}$
무승부	$0 \le \text{Output} \le 1$	$Output = \begin{cases} 0 & \text{승부가 날때} \\ 1 & \text{무승부} \end{cases}$
B 구단 승	$0 \le \text{Output} \le 1$	$Output = \begin{cases} 0 & B\text{구단 패} \\ 1 & B\text{구단 승} \end{cases}$

Output layer 의 Node 는 A 구단 승리, 무승부, B 구단 승리, 이렇게 3 개의 Node 로 구성하였다. 1982 년도부터 2014 년도까지의 시즌 별 데이터의 Output 값은 $0 \le out \le 1$로 나타나고, 2015 년도 시즌의 일일 경기 데이터의 Output 값은 A 팀이 승리면 1, 패배면 0 으로 설정하였다.

D. 학습 알고리즘

1982 년부터 2014 년까지의 데이터를 Neural Network 에 학습시키기 위해 Output 값과 Target 값의차를 오차(Error=Output 값-Target 값)로 하여, 그 오차에 비례시켜 Output Layer 의 weight 를 갱신하고, 그 다음으로 Hidden Layer 의 weight 를 갱신한다. 이 weight 를 갱신하는 방향은 Neural Network 의 처리방향(Feed-forward Propagation)과는 반대이다. 따라서 이러한 학습방법을 Back Propagation Algorithm(역-전파 알고리즘)이라 한다.

a. Feed-forward Propagation

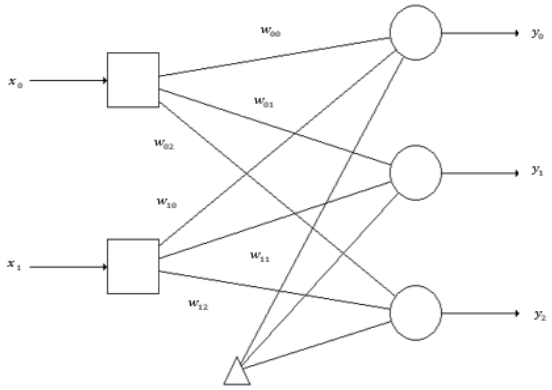

Figure 2. Neural Network

$$y_k = \sum_{j=0}^{m} w_{jk} x_j \qquad (1)$$

Feed-forward Propagation 알고리즘은 데이터들의 관계에 관해 학습된 내용들을 상위 계층으로 보내는 것이다. 각 Node 에 입력 값(x)들과 각 연결선의 가중치(w)를 특정한 수학적 함수를 통하여 변화해 다음 Layer 에 입력 값을 넣어주는 알고리즘이다[1][3].

b. Back Propagation

학습을 하기 위해서는 입력 데이터와 원하는 Target 데이터가 있어야 한다. 입력이 신경망의 가중치(weights)를 곱하고 더하는 과정을 몇 번 반복하면 입력의 결과 값인 Output 이 나온다. 이때 Output 은 학습 데이터에서 주어진 원하는 Target 과 다르다. 결국, 신경망에서는 (Output - Target)만큼의 오차(Error = Output - Target)가 발생하며, 오차에 비례하여 Output Layer 의 weight 를 갱신하고, 그 다음 Hidden Layer 의 weight 를 갱신한다. weight 를 갱신하는 방향은 신경망의 처리 방향과는 반대이다. 이런 이유로 Back Propagation 알고리즘이라고 한다[3].

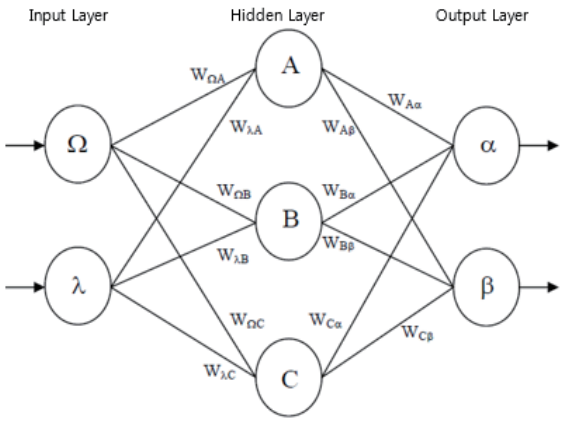

Figure 3. Neural Network

Output 의 Error 를 계산한다.

$$\delta_\alpha = out_\alpha(1-out_\alpha)(target_\alpha - out_\alpha) \qquad (2)$$

$$\delta_\beta = out_\beta(1-out_\beta)(target_\beta - out_\beta) \qquad (3)$$

Output Layer 의 weight 를 바꾸어준다.

$$\begin{aligned} w^+_{A\alpha} &= w_{A\alpha} + \eta\delta\alpha out_A \\ w^+_{A\beta} &= w_{A\beta} + \eta\delta\beta out_A \\ w^+_{B\alpha} &= w_{B\alpha} + \eta\delta\alpha out_B \\ w^+_{B\beta} &= w_{B\beta} + \eta\delta\beta out_B \\ w^+_{C\alpha} &= w_{C\alpha} + \eta\delta\alpha out_C \\ w^+_{C\beta} &= w_{C\beta} + \eta\delta\beta out_C \end{aligned} \qquad (4)$$

Hidden Layer 의 Error 를 계산한다.

$$\begin{aligned} \delta_A &= out_A(1-out_A)(\delta_\alpha w_{A\alpha} + \delta\beta w_{A\beta}) \\ \delta_B &= out_B(1-out_B)(\delta_\alpha w_{B\alpha} + \delta\beta w_{B\beta}) \\ \delta_C &= out_C(1-out_C)(\delta_\alpha w_{C\alpha} + \delta\beta w_{C\beta}) \end{aligned} \qquad (5)$$

Hidden Layer 의 weight 를 바꾸어준다.

$$\begin{aligned} w^+_{\lambda A} &= w^+_{\lambda A} + \eta\delta Ain_\lambda \\ w^+_{\Omega A} &= w^+_{\Omega A} + \eta\delta Ain_\lambda \\ w^+_{\lambda B} &= w^+_{\lambda B} + \eta\delta Bin_\lambda \\ w^+_{\Omega B} &= w^+_{\Omega B} + \eta\delta Bin_\lambda \\ w^+_{\lambda C} &= w^+_{\lambda C} + \eta\delta Cin_\lambda \\ w^+_{\Omega C} &= w^+_{\Omega C} + \eta\delta Cin_\lambda \end{aligned} \qquad (6)$$

여기서 η 는 보통 1 을 사용하며, 학습 속도를 높이거나 늦추는 역할을 한다.

Ⅲ. 실험 결과

A. 예측 에러 분석

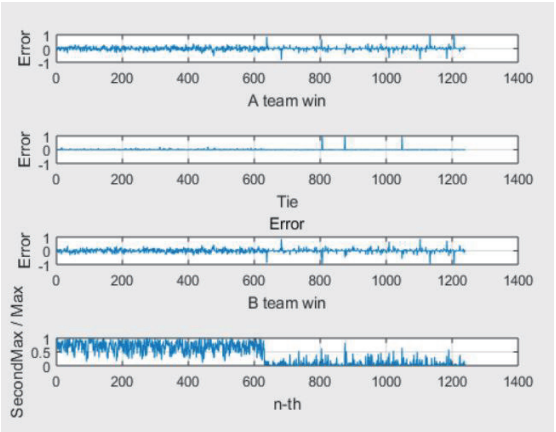

Figure 4. 학습결과의 Error

Input 을 통해 학습시킨 모델을 통해 Output 을 구하였

고 Target 과 비교를 하여 error 를 구했다.

B. 예측 에러의 상관성

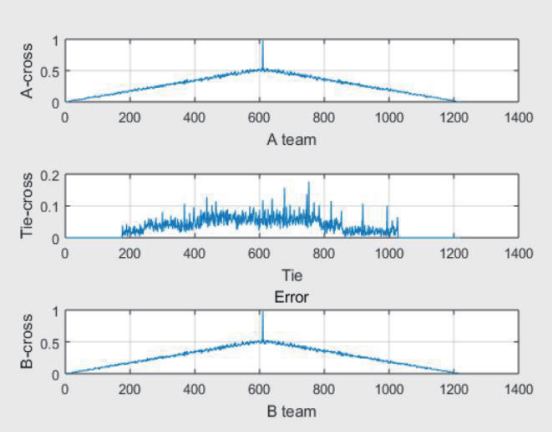

Figure 5. Target 과 Output 의 Cross-Correlation

Target 과 Output 의 Cross-Correlation 을 통해 Output 이 잘 나왔는지 확인했다.

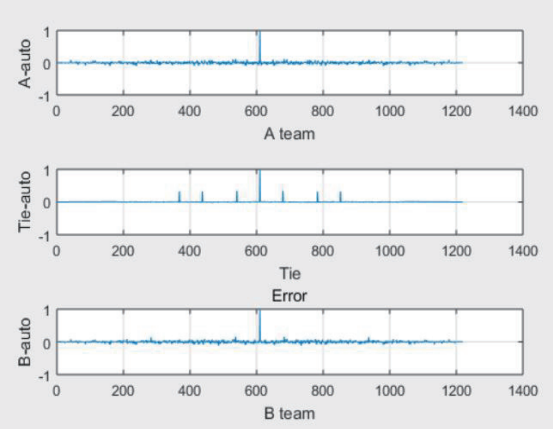

Figure 6. 발생 에러신호의 Auto-Correlation

Target 의 Auto-Correlation 을 통해 Output 이 잘 나왔는지 확인했다.

C. 예측된 프로야구 승률 결과

Table 4. 예측승률과 실제승률

구단	예측승률	실제승률	구단	예측승률	실제승률
삼성	61.7%	60.3%	두산	48.2%	54.3%
넥센	53.9%	54.3%	한화	47.5%	47.5%
NC	52.5%	59.4%	SK	45.3%	48.9%
KIA	51.8%	47.5%	LG	42.9%	44.9%
롯데	49.6%	46.4%	KT	41.8%	36.4%

9 월 30 일을 기준으로 각 구단의 승률과 미리 예측한 승률을 비교해본 결과 각 구단의 승률과 예측승률이 평균 3.39%의 차이가 났다.

Ⅳ. 결론

본 논문에서는 Neural Network 를 사용하여 야구경기의 승패를 예측하는 모델을 구현하였다. 과거의 데이터들을 사용하여 미래 경기의 승패를 예측하였다. 9 월 6 일부터 9 월 30 일까지의 야구경기를 예측한 값과 실제 경기 결과값을 비교하였다. 그 결과 각 구단의 승률과 예측승률이 3.39%의 차이가 났다.

참 고 문 헌

[1] S. Y. KUNG, Digital Neural Network, Prentice Hall, 1993

[2] SIMON HAYKIN, Neural Networks: A comprehensive Foundation, IEEE Computer Society Press, 1994

[3] JACEK M. ZURADA, Introduction to Artificial Neural Systems, West Publishing Company, 1992

[4] Howard Demuth, Mark Beale, MATLAB: Neural network toolbox: user's guide, MathWorks, Inc.,1994

[5] Judith E. Dayhoff, Neural network architectures: an introduction, N.Y.: Van Nostrand Reinhold, 1990