5-2022

Deep Learning in Sports Prediction

Fan Lee
Trinity University, fanlee916@gmail.com

Deep Learning in Sports Prediction

Fan Lee

Abstract

Sports prediction has always been an interesting problem in the entertainment industry. Many data scientists have come out different methods on this problem. We hope to see how well a neural network model can predict an individual game outcome and the final ranking on NBA data. We examined the possibility of different unbiased deep learning models can perform as well as other mathematics methods. We were also looking for what types of data are more influential for the models. Then, we can make some assumptions on our models and the other sports prediction methods.

Acknowledgments

First, this thesis cannot be completed without my parents, Dr. Chia-Yen Lee and Hsun-Yu Chang who financially support me from Taiwan.

Second, I would like to thank Dr. Hibbs who has guided me throughout the entire thesis. Dr. Balreira also inspired me a lot and helped me on the mathematics part of the research. Also, I want to thank Dr. Lewis and Dr. Jiang who helped me on building the fundamental understanding on machine learning at Trinity University.

Deep Learning in Sports Prediction

Fan Lee

A departmental senior thesis submitted to the
Department of Computer Science at Trinity University
in partial fulfillment of the requirements for graduation
with departmental honors.

April 19, 2022

Thesis Advisor

Department Chair

Associate Vice President
for
Academic Affairs

Deep Learning in Sports Prediction

Fan Lee

Contents

List of Tables

List of Figures

Chapter 1

Introduction and Foundational Information

This chapter will introduce what inspired us to do this research.

1.1 Sports Markets

Sports markets have grown a lot in the last few decades. Some of the professional sports leagues even have billions of dollars in revenue each year, such as NBA, NFL, and MLB. Winning a match is the most important part, so being able to accurately predict the sport result is always a challenging but interesting problem. Indeed, the unpredictable outcome is one of the main reasons that people love playing and watching sports. The stakeholders who are interested in predicting sports results include bookmakers and sports betting platforms, as well as gamblers who bet on match results. According to Grand View Research[25], the global sports betting market was valued at USD 66.98 billion in 2020, and an Esports betting company, LOOT.BET, observed over 67 percent growth in online betting volumes. Sports

result prediction is also potentially useful to players, team management and performance analysts in identifying the most important factors that help to achieve winning outcomes, upon which appropriate tactics can be identified[7].

1.2 Overview of NBA

Many betting companies and sports websites like ESPN predict the game outcome of National Basketball Association (NBA) because it is the second largest sports league in the world with 6.41 billion USD revenue in 2021[15]. Due to the time constraint, this thesis will only focus on NBA regular season prediction which consists of 30 teams in the US and Canada. Half of the teams are in the East division and the other half are in the West division. Each team plays a total of 82 games in a regular season. There is no tied game in basketball, so if two teams are still tied after 4 quarters of 12 minutes regular time, there will be an extra 5 minutes of overtime. At the end of the season, 8 out of 15 teams with the most wins in each division can advance to the playoffs. An NBA season usually goes across two years, and we used the starting year as the season name in this thesis.

1.3 Discussion

Many of the existing sports prediction methods use preset parameters which require an expert of that sport to do data analysis for setting the parameters. However, it is time-consuming, and any human being can be biased when doing this. Many other math methods might not do so well if there is not much data to be used. So, our goal is testing out the possibility of building an unbiased deep learning model for the same topic and trying to take the advantage of this machine learning method. In addition, we want to argue that the existing ranking method may not be perfect because it sorts the team strength solely

based on counting their win-loss record. The higher ranking team does not always have higher odds to win the match. A fair power ranking method should be able to be used for sorting the strength of teams in a sports league.

Chapter 2

Related Works

This chapter will discuss the related work of game prediction methods and deep learning.

2.1 Sports Prediction Methods

Predicting the result of sports games has been a common problem for over a hundred years[14]. This section provides a few sports prediction methods made by the scientists for an individual game outcomes.

2.1.1 Elo Method

The Elo system is a mathematics method which was first used for chess games to determine the strength of a player[3]. Each player is given an initial rating y. When player i plays player j, the ratings of both players are updated using a function E that depends on their match outcome O (Win, Tie or Loss). Generally the winner will gain a certain amount of points from the loser or vice versa, so the sum of all ratings remains constant and the average of the league is always the same. Equation (2.1) and (2.2) shows a general idea of Elo. The

parameters of function E vary from different sports and model designers. The method is quite well-known, and it is still used by some popular websites like FiveThirtyEight.

$$y_i \leftarrow E(y_i, y_j, O_{ij}) \tag{2.1}$$

$$y_j \leftarrow E(y_j, y_i, O_{ji}) \tag{2.2}$$

2.1.2 Colley's Method

Wesley Colley created the Colley's Method in 2002 for computing ratings of n number of teams in competitive sports. The only information used by this model are wins and losses for each team. As seen in equation (2.3), matrix M where M is an i by j matrix and B is an i by 1 matrix. The (i, i) entry of M is two plus the number of games played by T_i for each i, and the (i, j) entry of M is the negative of the number of games played between T_i and T_j for $i \neq j$. The entry B_i is in equation (2.4), where w_i is the number of wins by T_i and l_i is the number of losses by T_i. We then let the i-th entry of the exact solution X be the rating of T_i. In Figure 2.1, team i's rating depends on the ratings X_i of all its opponents[9][22]. The concept of Colley's Method is a team should get more points when beating a stronger team than a weaker team and it is unaffected by differences in the final score[16][26].

$$M_{nn}X = B \tag{2.3}$$

$$B_i = 1 + \frac{w_i - l_i}{2} \tag{2.4}$$

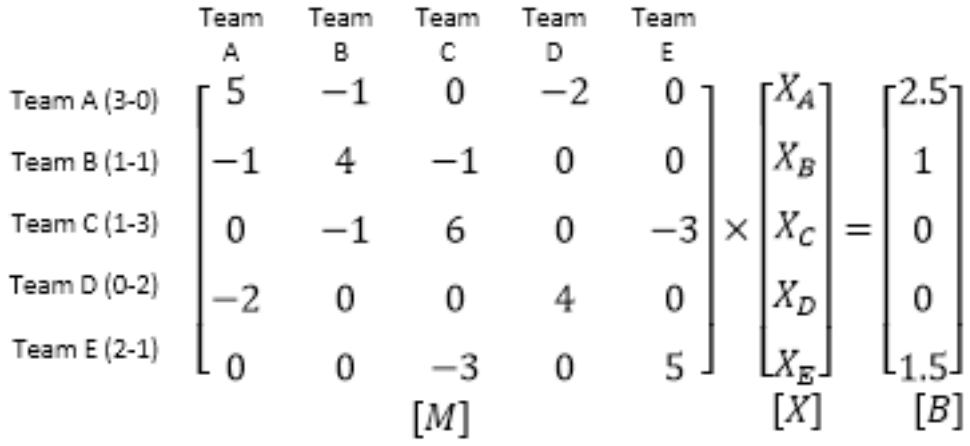

Figure 2.1: Colley's Method Example

2.1.3 The Oracle Method

The Oracle Method is an advanced Markov method[10] that has the ability to consider multiple teams' traits at once. In a league of n teams, creating a matrix which has n by n matrix and each node is determined by the win loss record. This idea is a generalization of the personalization vector in the PageRank method[18] that addresses a fundamental flaw when using certain Markov methods to rank a tournament. The advantage of the method is even there were not many games have been played yet, the model can still generate the ranking. The method is have an extra row and column for the n by n matrix where n is the number of the teams, so it can be written as equation (2.5). When an undefeated team loses to a winless team, and subsequently, that previously winless team will rise up to near the top of the rankings. The Oracle will be a new, $(n+1)$-st node in the network, N^m, associated to the tournament. Then, in the new network, N_O^m, to be the network N^m with a new node, $n+1$, that has a directed edge to and from node $n+1$ to each node 1,2,...,n. This new node changes the associated n×n matrix A^m into an $(n+1)$ by $(n+1)$ Oracle matrix,

O^m. Using equation (2.5) to compute the transitional probabilities, we have that equation 2.6 where l_i is the number of losses by T_i. Then, the Perron-Frobenius Theorem[24] can always be applied to find an Oracle rating vector $r^O \in R^{n+1}$, and then define the rating vector $r \in R^n$ in equation (2.7)[4]. Figure 2.2 shows a graph explanation.

$$O_m = \begin{bmatrix} A^m & e \\ e^T & 0 \end{bmatrix} \tag{2.5}$$

$$p_{n+1,i} = \frac{1}{l_i + 1}, p_{i,n+1} = \frac{1}{n} \tag{2.6}$$

$$r_i = \frac{r_i^O}{\sum_{j=i}^{n} r_i^O} \tag{2.7}$$

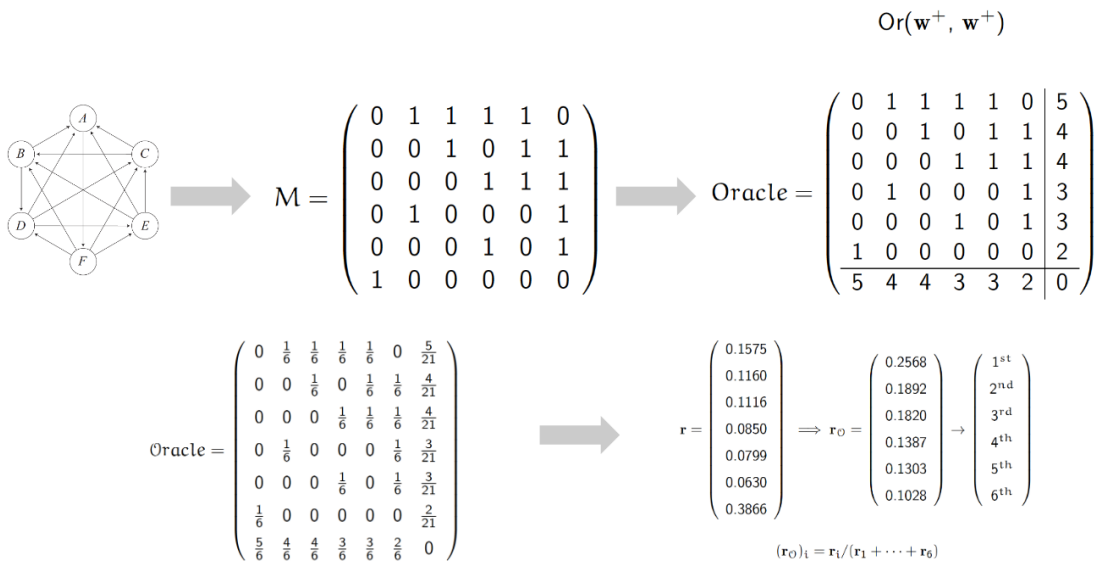

Figure 2.2: The Oracle Method Example

2.1.4 Linear Regression Method

Linear regression is commonly used in many fields, including engineering, physics, and chemistry. Equation (2.8) is called a simple linear regression model. Customarily, x is called the independent variable or regressor variable and y is called the dependent variable or response variable. The difference between the observed value of the left side of the equation and the right side of equation is ϵ. It is a random variable that accounts for the failure of the model to fit the data[21][21]. When it is used in sport predictions, it is built multiple linear regression model that takes more dimensions in the response variable for evaluating the strength of a team[23]. It can be written as a matrix equation to do the multiple linear regression[12]. In equation (2.9), it can take n variables.

$$y = B_0 + B_1 x + \epsilon \tag{2.8}$$

$$y = B_0 + B_1 x_1 + B_2 x_2 + B_3 x_3 + ... + B_n x_n + \epsilon \tag{2.9}$$

2.1.5 Massey's Method

Massey's method uses a linear least squares regression to solve a system of linear equations. It can be written as equation (2.10) where M is a i by j matrix, r is the unknown rating vector, and p is a vector of cumulative point differentials. M_{ij} is the number of games teams i and j played, multiplied by -1 and M_{ii} is the total number of games team i has played. Then getting the least square solution for M where r is the rating for each team[26][17].

$$Mr = p \tag{2.10}$$

2.1.6 Decision Tree Method

A decision tree recursively separates observations into branches for the purpose of achieving the highest possible prediction accuracy. In order to split the pool of observations into two or more subgroups by an algorithm, it is repeated at each leaf node until the entire tree has been constructed. The classification and regression trees which were first developed by Breiman, Friedman, Olshen, and Stone[6] are commonly used for sports prediction[11].

2.1.7 Limitations and Discussion

Some of the methods such as Elo require the data scientists to preset the parameters of the equation for the sport. Also, the adaptation of a new trend can be slow and the human-setting of parameters can be biased. Some methods like Massey's method does not perform

well if there are less games than the number of the teams. Furthermore, many of these methods only takes win, loss outcome as input data, so they might ignore the fact that there might be a trend in other data input. All these methods are not possible to build a play-by-play game prediction model for the same reason. Some of these issues are possible to be tested out by implementing deep learning methods.

2.2 Deep Learning

Deep learning is obtained by composing simple but non-linear modules that each transform the representation at one level into a representation at a higher, slightly more abstract level. With the composition of enough such transformations, very complex functions can be learned[19].

2.2.1 Overview

The basic concept of neural networks (NN) can be dated back to 1943, and it has been widely used in recent decades along with better hardware and more research has been done[20] . NN is a type of machine learning and has become relatively competitive to conventional regression and statistical models accuracy when used for classification, clustering, pattern recognition and prediction in many disciplines. NNs are mostly used for universal function approximation in numerical paradigms because of their excellent properties of self-learning, adaptivity, fault tolerance, nonlinearity, and advancement in input to an output mapping[1]. A NN model like Figure 2.3 consists of several layers of nodes and each node in a layer is connected to all the nodes in later layer with a weight function. There may be a activation function in the neurons so the model can be more stable inside the constraint. After passing the input data into the network, the variables will do several matrix multiplication, and

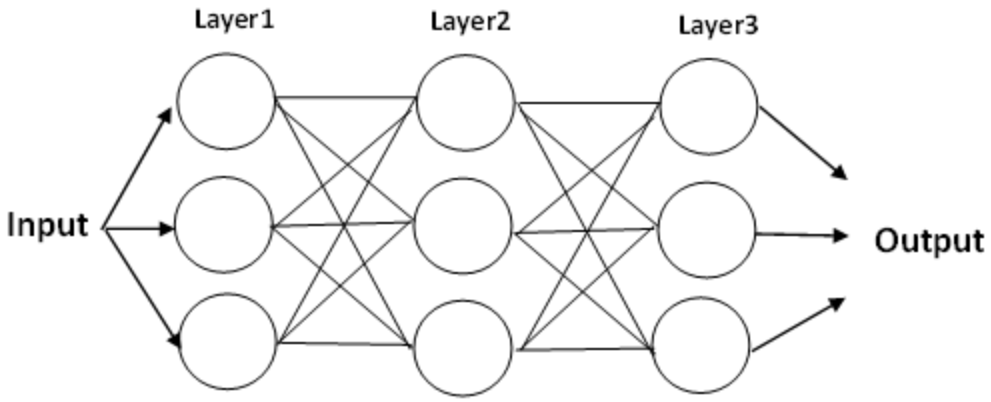

Figure 2.3: A general NN

output a fixed-sized data. Then, we can use a loss function to get the difference between the output and the label and then use backpropagation algorithm to update the parameters of the network. There are several variants of NN. Convolution Neural Networks (CNN)[2] are commonly used in image recognition and speech recognition. Recurrent Neural Networks (RNN)[27] are commonly used in natural language processing by utilizing the advantage of taking a sequence of inputs.

2.2.2 Challenges

In order to train an typical NN model, we need a huge amount of data. Also, each training will consume a considerable amount of computing power and has high time complexity, so it is only recommended to run on high performance GPU. The down side of fully machine made models is that it is hard for humans to understand the significance of all the parameters in the model. It is almost like a black box[8], so we can only see garbage in garbage out[5] during and after training a model.

Chapter 3

Methods

This chapter will discuss the details of how we built and trained our models. We used the NBA team stats data and play-by-play data collected by National Statistical for season 2014-2019. We chose not to use season 2020 because that season was impacted by COVID, and the season was shorter than others. The methods we used to collect our result are also included here.

3.1 Open Source Software Used

We chose Google Colab to be our environment, and we carried out our project fully based on Python 3 and Pytorch library. For data visualization, we mainly used matplot library on Python 3.

3.2 Neural Network Implementation Details

This section will briefly explain some details of the neural network we used in the thesis.

3.2.1 Terminology of NN

Here are a few terms that machine learning scientists like to use.

Tensor
Tensor is a multi-dimensional matrix containing elements of a single data type.
Epoch
The number of times we passed the whole training dataset.
Batch
The number of data taken to update the model parameters.
Weight
The parameters of the functions between two nodes.
Learning rate
A number which determines how much the weight will be updated.

3.2.2 Neural Network

We utilized artificial neural networks (NN) as our main machine learning method. An NN model in Figure 2.3 consists of several layers of nodes. Each node in a layer connected to all the nodes in another layer with a weight function. After passing the input data tensor into the network, the variables will do several matrix multiplication and then output a fixed-sized output tensor. There is also an activation function in the neurons to keep the number in constraint for stabler performance. We chose tangent as our activation function. During the training phase, testing labels are used to determine how far from the correct solution the output predictions are. A loss function is used to quantify this difference, and then after a backpropagation algorithm is applied to update the parameters to lower the loss. After our model has been through enough training, then we can test it with our separated test

dataset to determine the performance.

3.2.3 Loss Function

The loss function is crucial during training a model. The loss function determines how much the backpropagation step will update the weight parameters. We use Negative Log Likelihood Loss (NLLLoss) (3.1) for binary (win-loss) labels. For the point spreed models, we use Mean Square Error Loss (MSELoss) (3.2) because it is more suitable to continuous (non-binary) outputs.

$$L(x) = -log(x) \tag{3.1}$$

Negative Log Likelihood Loss Function

$$L(y, \hat{y}) = \frac{1}{N} \cdot \sum_{i=0}^{N} (y - \hat{y}_i)^2 \tag{3.2}$$

Mean Square Error Loss

3.2.4 Optimizer

An optimizer is the specific algorithm used to update the attribute weights of the network. All parameters passed to the optimizer are retained inside the optimizer object so the optimizer can update their values and access their grad attribute which is for calculating back propagation[13]. The two optimizer functions we used were Stochastic Gradient Descent Optimizer(SGD) and Adaptive Moment Estimation Optimizer(Adam). SGD performs a parameter update for each training example. Adam optimizer updates the learning rate for each network weight individually. It has faster running time, low memory requirements, and requires less tuning. These two are commonly used optimizers for these two kinds of

NN, so we did not try to use other optimizers.

3.2.5 Recurrent Neural Network

A Recurrent Neural Network (RNN) is a variant of NN which has another hidden layer connected to the input data and layer 1. The hidden layer will be passed down for another data input. The advantage of RNN is it can process input of any length, so the computation takes into account historical information by sharing the same hidden layer across time.

3.3 Predicting Individual Game Outcome *a priori*

There are multiple types of models we built to compare the accuracy of predicting the winner of a single game by using the team information prior to that game. Each tensor contains the information of both upcoming teams. All of them contain the data from the previous four months of games in the same season, so the testing data only contains games of the last two months. For each game day, we trained a new model by feeding all the previous games data as inputs and tested it with the new game's outcome. Each season we built about 50 models and averaged across them all. We always moved the home team data in the first half of the tensor in order to represent the home court advantage (Table 3.1). For consistency, the neural network models all contain three linear layers, and all interior nodes utilize a hyperbolic tangent activation function. Training was performed with a batch size of 4 and a learning rate of 0.005 over the course of 30 epochs. By our experiments, a small change of the learning rate and epoch did not cause any significant difference in accuracy, so we stuck we these two numbers for a fair comparison. There are two types of labels we tried; one is "home team win/loss label" encoded in 1 or 0 and the other is the "point spread label" which is the point difference at the end of the game. For all of our NN models, we

Year	2012	2013	2014	2015	2016	2017	2018
Home Team Winning Rate	0.612	0.581	0.575	0.589	0.584	0.579	0.592

Table 3.1: Home team winning rate by year

used SGD as our optimizer.

3.3.1 NN with all Data

(All-WL) We used all the dimensions of each game such as total points scored, 3 point attempts, 3 point shots made, and fouls as a team, and we added the average N previous games data for both teams in this method, including their winning rate before the game (see Table 3.2). Also, we used one-hot encoding for each unique team ID. Home court advantage and the date of the game are also passed in the tensor. We normalize our data because it will be more stable during training. The dimensions of each input tensor are 112. It has the standard 3 layers : $(112, 64), (64, 16), (16, 2)$ and a log soft max function for the output, so the output is a single variable. The NLLLoss was used for this network (Figure 3.1).

3.3.2 NN with Winning Record, and the Unique Team ID

(TIDWR-WL) We tried to determine how recent game data can affect a team's performance, so we changed the window size on averaged stats. However,we found that changing the window size of averaged stats of the previous method causes no significant differences in performance, so we decided to not use the individual stats data for better speed performance. Without the average stats, we have home court advantage, date, and team ID. The size of input tensor is 82 in this method. The three layers are $(82, 64), (64, 16), (16, 2)$. It also has

input data	number of val	range of val	example	encoding
day of the season	1	0-300	65	norm to 0-1
team ID	30 teams	0-1	1	one-hot encoded
total wins	1 per team	0-72	35	norm to 0-1
total losses	1 per team	0-72	15	norm to 0-1
wining rate	1 per team	0-1	0.4	
minutes	1 per team	0-260	235	norm to 0-1
total points	1 per team	0-160	95	norm to 0-1
total field goals made	1 per team	0-60	40	norm to 0-1
total field goals attempt	1 per team	0-120	80	norm to 0-1
total 3 points made	1 per team	0-30	15	norm to 0-1
total 3 pints attempt	1 per team	0-70	31	norm to 0-1
total free throw made	1 per team	0-50	14	norm to 0-1
total free throw attempt	1 per team	0-60	27	norm to 0-1
total rebounds	1 per team	0-80	45	norm to 0-1
total assists	1 per team	0-50	25	norm to 0-1
total steals	1 per team	0-20	7	norm to 0-1
total blocks	1 per team	0-20	8	norm to 0-1
total offensive rebounds	1 per team	0-50	15	norm to 0-1
total turnovers	1 per team	0-50	18	norm to 0-1
total fouls	1 per team	0-50	21	norm to 0-1

Table 3.2: Input data

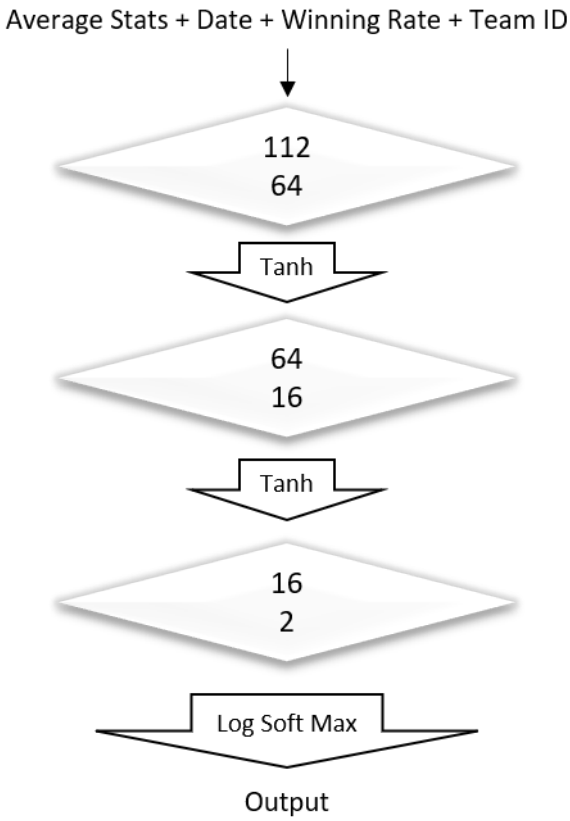

Figure 3.1: NN with average stats, winning rate, and the unique team ID figure

log soft max at the end and NLL as the loss function.

3.3.3 NN Predicting Win Loss with Team ID

(TID-WL) Several sports prediction methods based on matrices, such as the Oracle method and Massey and Colley method, use only win loss records. As such, we investigated if the neural network model with only the home court advantage, date and the team ID can affect the performance. The input size of this tensor is 78, and the three layers are $(78, 64), (64, 16), (16, 2)$. It also has log soft max and NLLLoss.

3.3.4 NN Predicting Point Spread with all Data

(All-Spread) Another way to label our data is by calculating the point spread which is the point difference of two teams at game end. We did not use log soft max function for Spread model outputs and we chose MSELoss (Mean Square Error) as our loss function, since we want to backpropagate by the scale of difference of label and prediction. We used all the dimensions of each game and we added the average N previous games data for both teams in this method, including their winning rate before the game and unique team ID.

3.3.5 NN Predicting Point Spread with Team ID

(TID-Spread) We only used the unique team ID for input data for comparison. We chose MSELoss as our loss function. The model structure is mostly like TID-WL, but the label is the point difference instead.

3.3.6 RNN Predicting Win Loss with all Data

(RNN-All-WL) In order to find trends during the season, we tried to implement a Recurrent Neural Network as our model. For each training data, we fed in 5 consecutive games of the

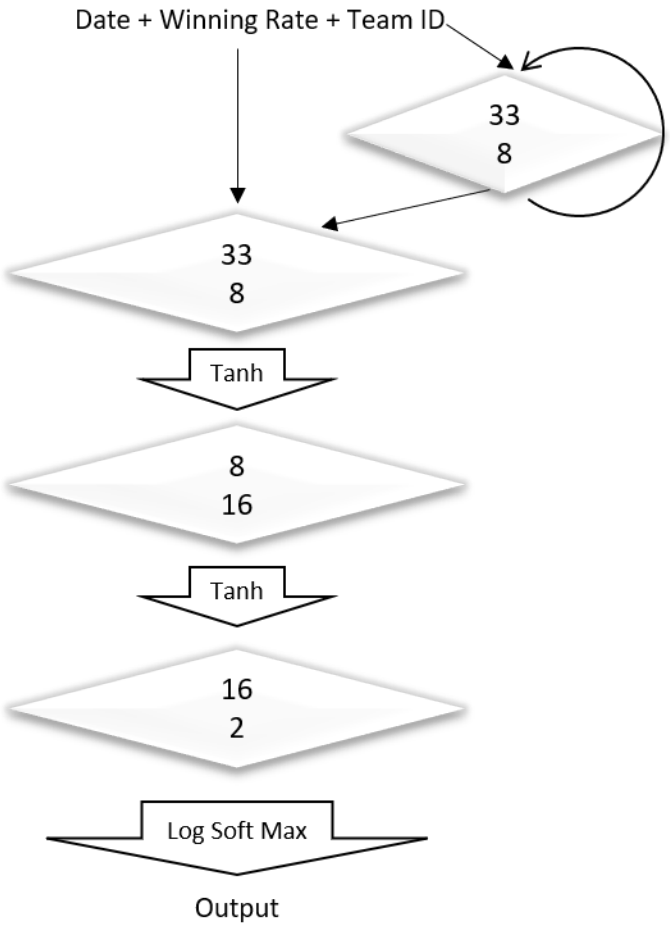

Figure 3.2: My RNN Figure

same team for one output. Each tensor has the game day, home court label, winning rate and the unique team ID. The size of the tensor is 25, and the hidden size is 8. It has three linear layers $(33, 8), (8, 16), (16, 2)$, and a hidden layer $(33, 8)$ so that every regression would have to pass in by multiple linear layers (Figure 3.2). We chose log soft max function and NLLLoss function. The label is home team win label which is 1 or 0, and the batch size is 1 here.

	All-WL	TIDWR-WL	TID-WL	All-Spread	TID-Spread
Average Stats	True	False	False	True	False
Winning Rate	True	True	False	True	False
Team ID	True	True	True	True	True
Input Tensor Size	112	82	78	112	78
Output Label	WL	WL	WL	Point	Point

Table 3.3: The comparison of our models

3.4 Prediction during Individual Games by Using Play-by-Play Data

(Play-by-Play) In order to predict the winner during a match, we used play-by-play data to build our NN models. Each play-by-play data contains different types of plays such as 2 point scores made, free throw attempts, turnovers, etc. We converted the type of plays into 1 one-hot encoding. Game date, time remaining, unique team ID, points at the time are also included. The size of each tensor is 110. It has three linear layers $(110, 64), (64, 16), (16, 2)$ with log soft max function, and the loss function is NLLLoss. The optimizer is SGD. The label is home team win label which is 1 or 0, and the batch size is 32.

3.5 Prediction of the Season Outcome

For predicting the ranking at the end of the season, we built our NN models based on the All-WL and TID-WL model, and then simulated the games to attain the season outcome. We began the test from the first 5 game days to the last game day.

3.5.1 Types of Ranking

There are two types of ranking in our methods.

Power Ranking to Predict Final Season Ranking

(PR Final) We simulate all the combinations of the teams. Each team would have 58 games against all the other teams and both home and away. After simulating all the games, we would have the rankings of the best team to the worst team in the league.

Simulated Season to Predict Final Season Ranking

(Sim Final) We used the current win loss records and the simulation of the rest of the season schedule with our model to attain our prediction of the regular season ranking.

3.5.2 NN with Average Stats, Winning Rate, and the Unique Team ID

(All-Rank) We used all the dimensions of each game such as total points, 3 point attempts, 3 points made, and fouls as a team, and we added the average N previous games data for both teams in this method, including their winning rate before the game. Also, we used one-hot encoding for each unique team ID. Home court advantage and the date of the game are also passed in the tensor. The dimensions of each tensor are 112. It has the standard 3 layers : $(112, 64), (64, 16)$, and $(16, 2)$ and a log soft max function for the output, so the output is a tensor float variable. NLLLoss is the loss function for this. After we trained our model, we implemented our 3 ranking methods to collect the predicting ranking.

3.5.3 NN with Team ID

(TID-Rank) In this model we only use team ID and game day to optimize the speed of training. The input size of this tensor is 78, and the three layers are $(78, 64), (64, 16)$, and $(16, 2)$. It also has log soft max and NLLLoss. Then, we use our 2 ranking methods to get the prediction.

	All-Rank	TID-Rank	TID-Rank-Pre
Average Stats	True	False	False
Winning Rate	True	False	False
Team ID	True	True	True
Last Season Data	False	False	True
Tensor Size	112	78	79

Table 3.4: Comparison of data for All-Rank, TID-Rank and TID-Rank-Pre

3.5.4 NN with Team ID and Previous Season Data

(TID-Rank-Pre) In order to determine if additional data would improve performance at the beginning of the season, we use the team stats data for that season and the last quarter of data from the previous season to train our models. Each tensor contains the previous season label, team ID and game day. The input size of this tensor is 79, and the three layers are $(79, 64), (64, 16)$, and $(16, 2)$. It also has log soft max and NLLLoss. Then, we use our 2 ranking methods to get the prediction.

Chapter 4

Result and Discussion

This chapter will show the result we collected and discuss the possibility. The Elo rating we used are collected by FiveThirtyEight which cumulatively calculating the NBA Elo rating from season 1947 to 2020.

4.1 Predicting Individual Game Outcome *a priori*

Most of my NN models have a similar structure. They have size of N input, 3 layers, $(N, 64), (64, 16)$, and $(16, 2)$, and it is similar to Figure 3.1. The batch size was 4. The learning rate was 0.005 and 30 epochs. Our models performed better than the home win rating which means they can learn something more than the home team advantage from the training data.

The way we test our data for individual games is we want to test the last quarter games of the season. Let's say we want to test out x to n day of the season, then on day $i \in (x, n)$. We trained our model with all the data from day 1 to $i - 1$, and then test it with all the games in day i. We did every day from day x to n which means built $n - x$ models, and

then the successfully predicted games divided by all the games would be one data of our prediction accuracy which is the y axis.

4.1.1 All-WL with different average windows size

We made our All-WL to have different average windows size, 5, 10 and 20. In Figure 4.1, the result of the different window sizes in season 18-19 has no significant difference. So we want to conclude that the window size in this range of the average stats data has no strong effect on our NN model's result. It might imply that the length of the stats range and the stats for the previous games do not matter that much for our models.

4.1.2 All-WL, TIDWR-WL and TID-WL

From the result of Figure 4.1, we want to see how important the result for our models is. In Figure 4.2, we compare All-WL, TIDWR-WL and TID-WL for season 2018-2019. The result of these three has no significant difference, which shows that only having two teams' ID, home court advantage dimension and the date are enough for the NN to be trained well. Thus, the result shows that the stat data does not have a strong impact to the result.

4.1.3 RNN and TID-WL

In Figure 4.3, we have RNN and TID-WL results. We see that although RNN took averaging 6.8 times longer to train a model, the prediction rate is not better than the TID-WL. The problem can be the small data amount which led to under-fitting. Our model will

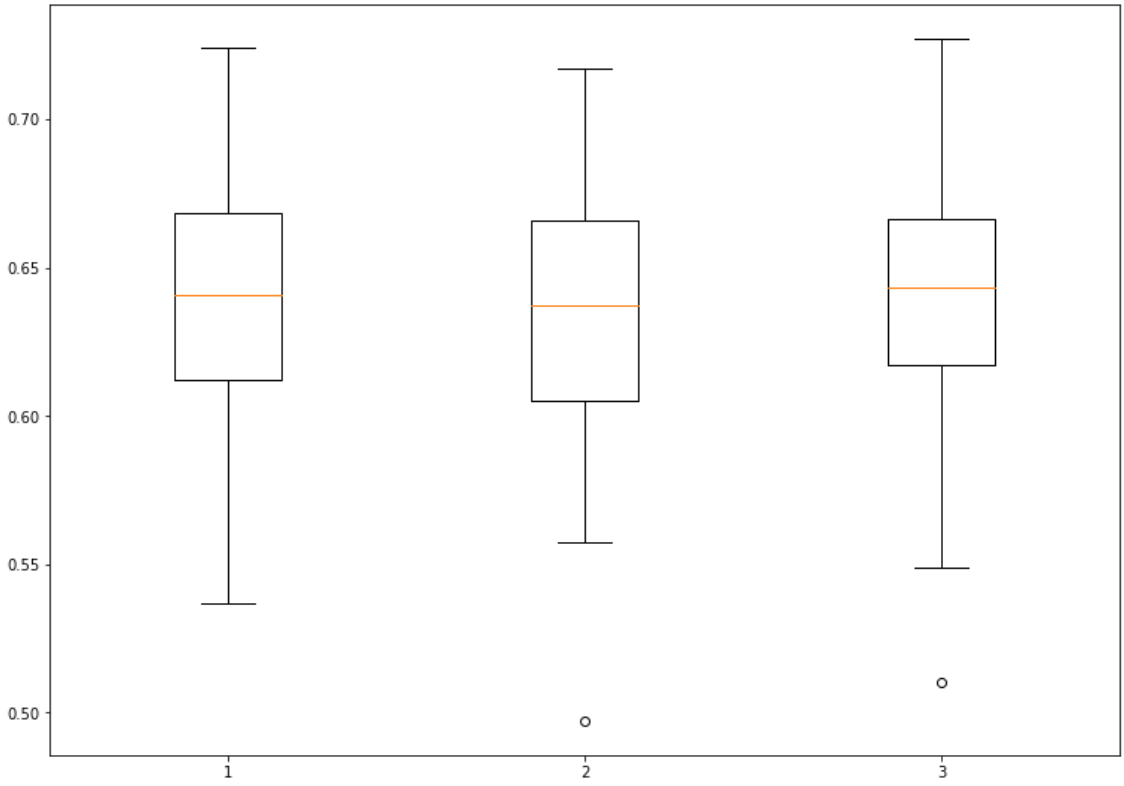

Figure 4.1: Boxplot of All-WL windows size prediction
1, 2 and 3 on the x-axis match to the windows size 5, 10 and 20
y-axis is the accuracy rate

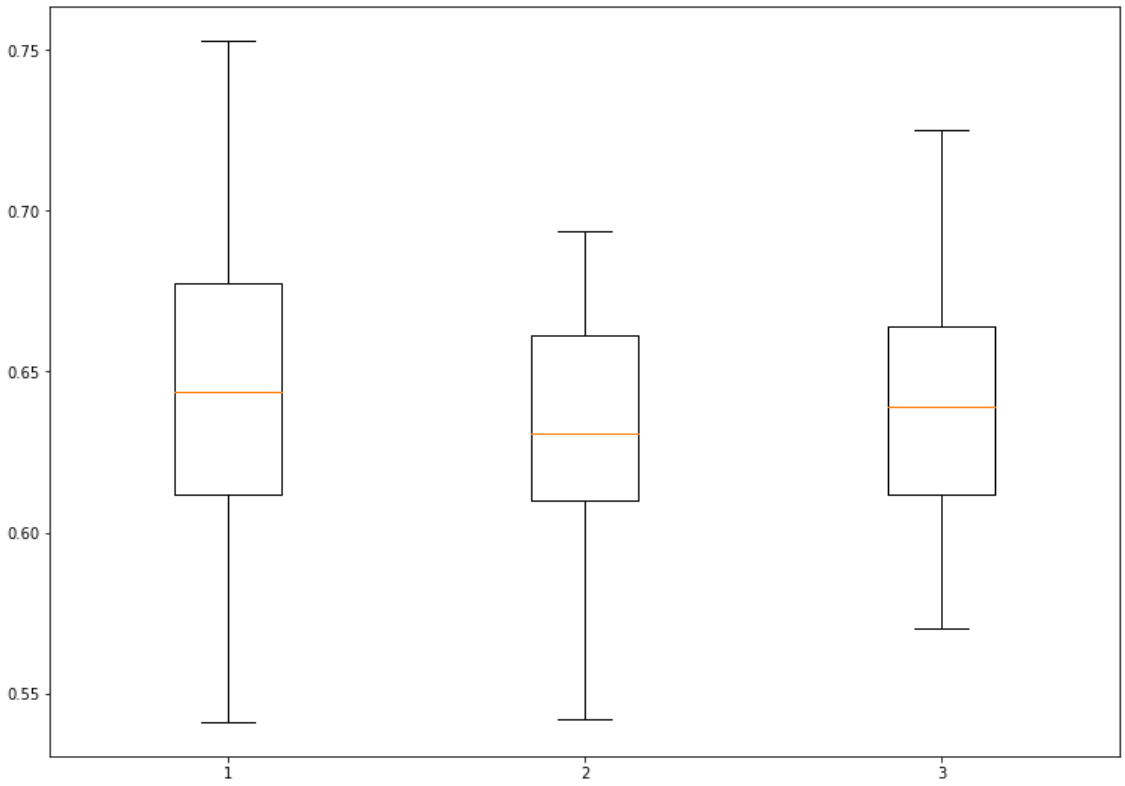

Figure 4.2: Boxplot of All-WL, TIDWR-WL and TID-WL
accuracy
1, 2 and 3 on the x-axis match to the models All-WL,
TIDWR-WL and TID-WL
y-axis is the accuracy rate

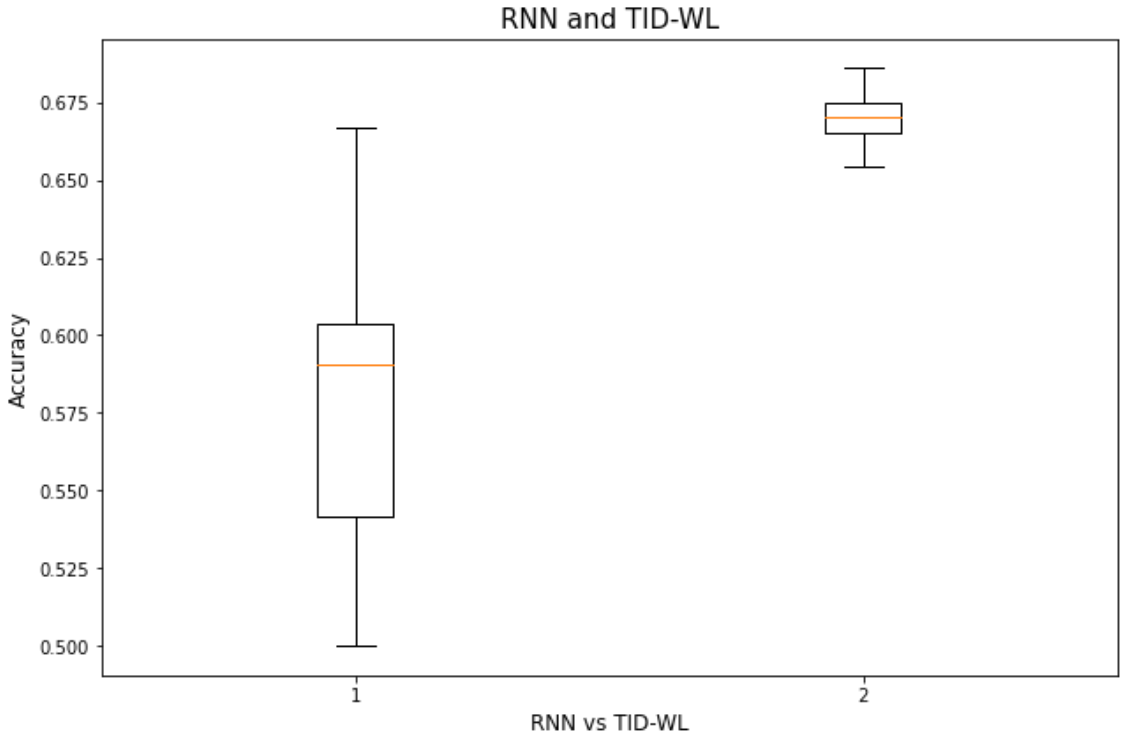

Figure 4.3: Boxplot of RNN and TID-WL accuracy
1 and 2 on the x-axis match to the models RNN and TID-WL
y-axis is the accuracy rate

take the data for a certain for its five previous games as input, but it led to the lacking of training size problem. Another reason could be our RNN structure does not math our need for prediction. The loss might vanish when doing backpropagation for multiple layers. By any means, the performance here for our basic structure RNN was not great, so for the same reason, a more complex model may not be helpful, and we do not move forward into building a more complex model. However, there is still a possibility that a good RNN model in this scenario existed, but we just do not know yet.

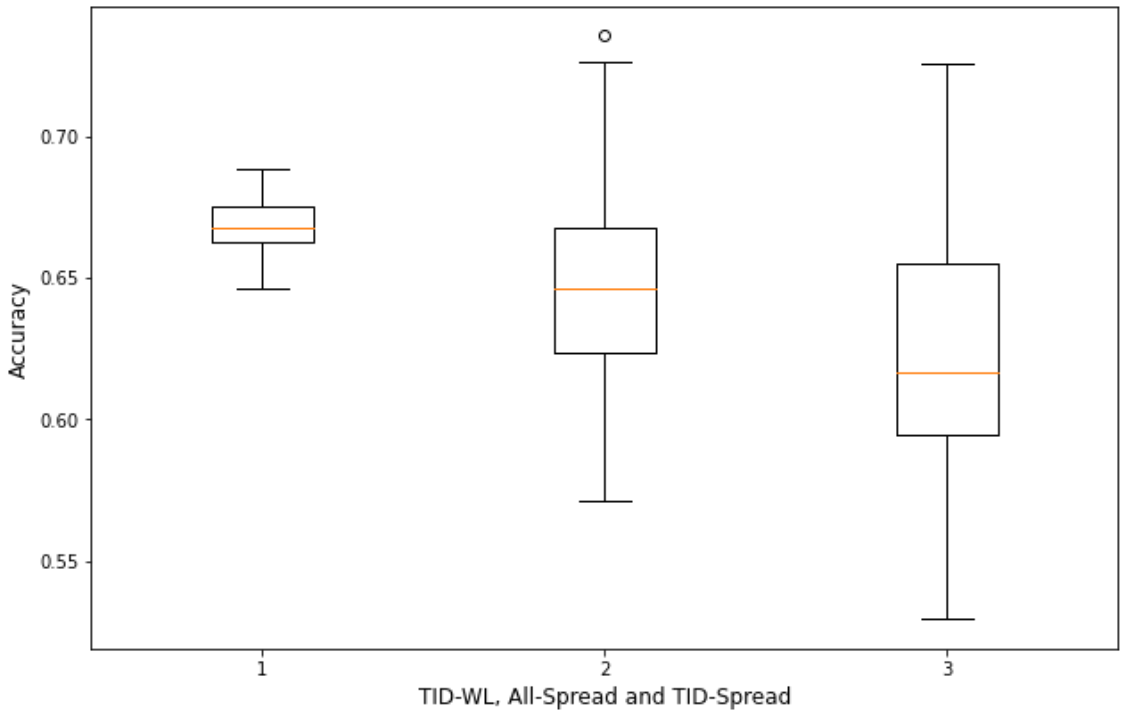

Figure 4.4: Boxplot of TID-WL, All-Spread and TID-Spread
accuracy
1, 2 and 3 on the x-axis match to the models TID-WL, All-Spread
and TID-Spread
y-axis is the accuracy rate

4.1.4 TID-WL, All-Spread and TID-Spread

In Figure 4.4, we also compare TID-WL, All-Spread and TID-Spread for season 18-19. We can see that the prediction of using win-loss label is more consistent and better than using point spread label. However, the All-Spread model is slightly better than the TID-Spread model which might imply that the machine can pick up some correlations between stats and point spread data if we are trying to predict the point spread of the game.

4.1.5 TID-WL for 5 seasons (2014-2018)

We can now conclude that our TID-WL model can achieve the best prediction rate and lower power usage by about 8 percent. In order to see if the method is robust across seasons, we check season 2014-2015, 2015-2016, 2016-2017, 2017-2018 and 2018-2019. Figure 4.5 shows that prediction rate of each season. We can see that the average prediction rate varies a lot in different seasons, but it also has a similar trend in the Elo method prediction Figure 4.6. It can be that there were more weaker teams who won the games in certain years. Also, our prediction rate is roughly as good as the Elo method model.

4.1.6 Discussion of the Results

The result shows that the NN models do not find helpful patterns within the extra average stats data. It is really interesting because it shows that the NN model was doing something like the Oracle method or Colley's method which only takes the win, loss and team as inputs but is able to give a rating for each team. Also, labeling the data on the point spread does not increase the prediction rate because we only care about predicting the winner here. We are guessing that the machine can build its own ranking method while being trained with team ID like a typical pure math method model. We can see that the best performance with the smallest input data is our TID-WL model.

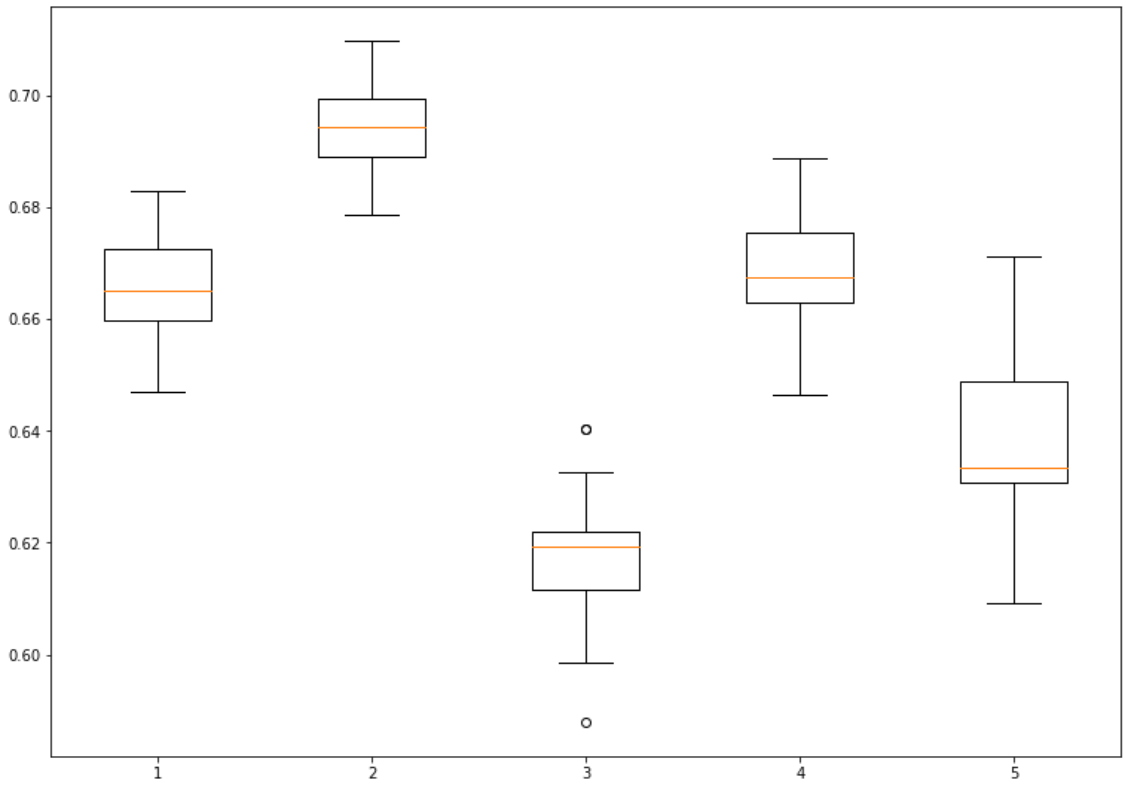

Figure 4.5: Boxplot of TID-WL for season 2014, 2015, 2016, 2017 and 2018
1, 2, 3, 4 and 5 on the x-axis match to the models TID-WL for season 2014, 2015, 2016, 2017 and 2018
y-axis is the accuracy rate

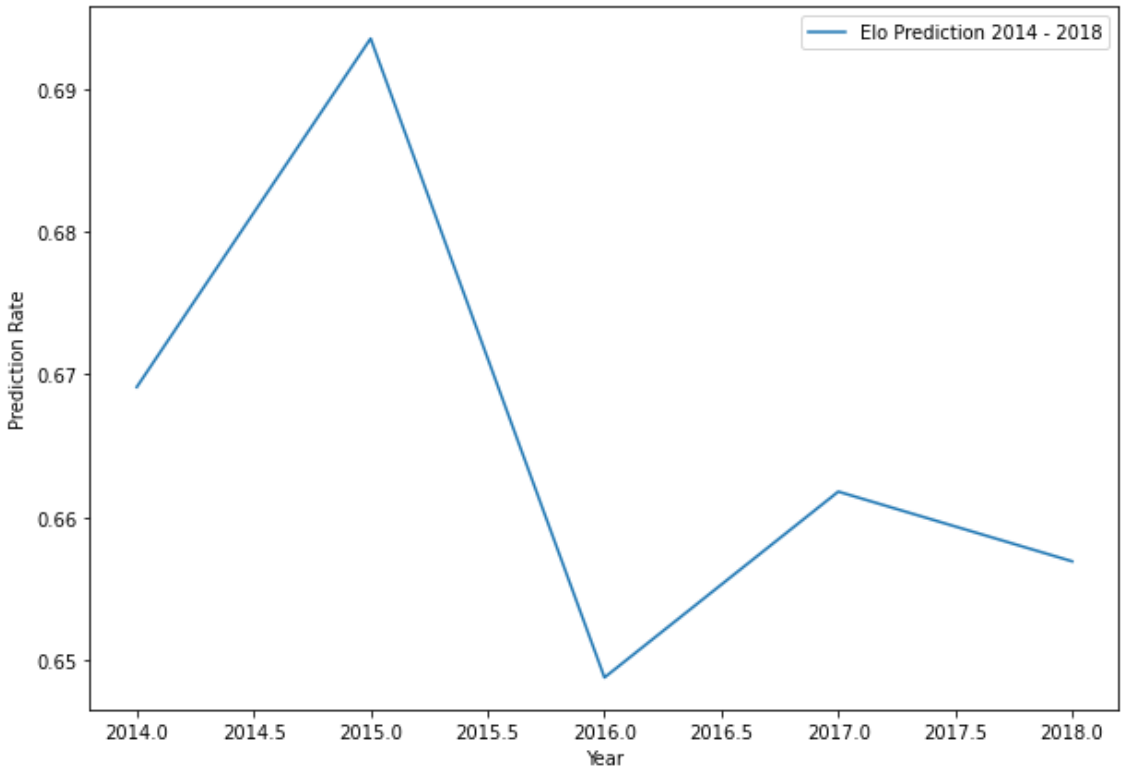

Figure 4.6: Line Graph of Elo Prediction Rate from season 2008 to 2018

4.2 Play-by-Play

We converted the type of plays into one-hot encoding dimensions. Game date, time remaining, unique team ID, points at the time are also included. The size of each tensor is 110. It has three linear layers (110, 64), (64, 16), (16, 2) with log soft max function, and the loss function is NLLLoss. The optimizer is SGD. The label is home team win label which is 1 or 0 and the batch size is 32.

Figure 4.7 shows the average of season 2018. The label of x axis is every 30 seconds of the game, the label of y axis is the average accuracy of the prediction at the time. The way we test our data is similar to individual game *a priori*, but instead of only has 1 label for each game, we have many data as all the plays of the game and we can averaging it into every 30 seconds.

We observed that the accuracy increase is closer to the end of the game which might be related to the average points difference and the time remaining. Our model does perform better when the time goes, and it passes the 63 percent accuracy which is our TID-WL accuracy for 2018 in the first few minutes. It shows that our model can actually learn from the play-by-play data.

4.2.1 Discussion of the Results

The prediction rate line in Figure 4.7 does not overlap with point difference line which means there are more factors than the points difference on the prediction and time. We did not compare this model to other math models because at the time we built this, there is no existing opened source model doing the same NBA play-by-play prediction model.

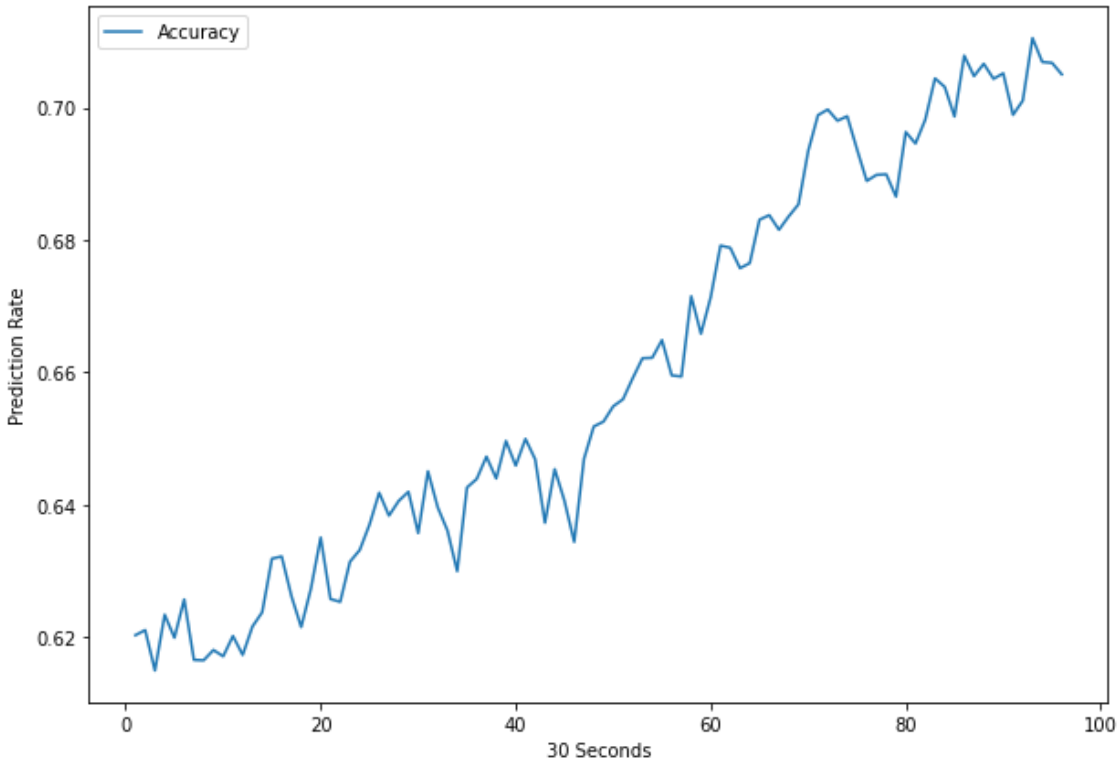

Figure 4.7: Line graph of Play-by-Play model in season 2018
(x-axis is 30 seconds, left y-axis is for accuracy)

4.3 Prediction of the Season Outcome

Our models here are either based on All-WL or TID-WL models. 3.1 After the models are trained, we simulated the season outcome and then used three types of testing methods on the models, which are PR Final, Sim Final, and Sim Playoffs. For PR Final and Sim Final, and we use the Kendall Tau function to compare the prediction to the real outcomes. We chose the Kendall Tau rank correlation coefficient to compare two rankings of list, and the value is determined by how many swaps are required for our ranking list to be the same as the real rankings.

4.3.1 All-Rank and TID-Rank

In Figure 4.8, our result shows that All-Rank and TID-Rank models do not have any significant difference, so we used the TID-Rank model for faster speed. Also, we just want to point out that Sim Final ranking method is more accurate because it inherit the record from before, so it will look more similar to the real final ranking.

4.3.2 TID-Rank and TID-Rank-Pre

For TID-Rank-Pre, the data set includes the last quarter of data from the previous season. We found out that in Figure 4.9 the model has more accurate prediction especially in the beginning of the season due to more training data.

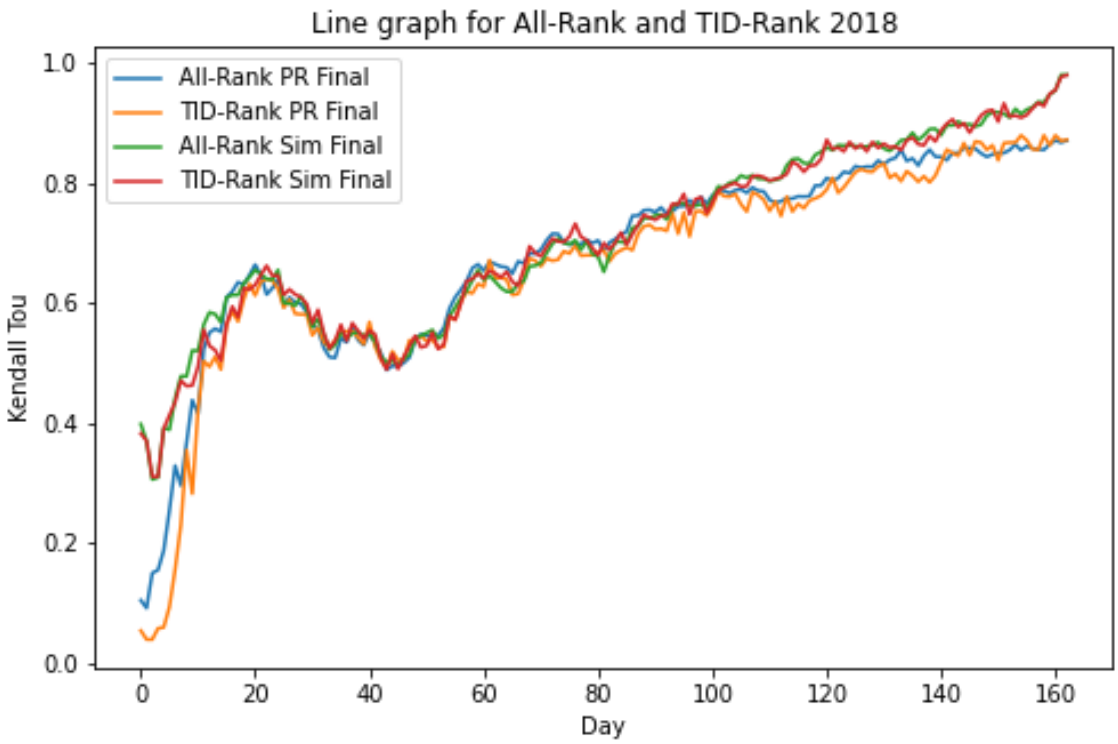

Figure 4.8: Line graph of All-Rank and TID-Rank for season 2018
x-axis is the day of the season
y-axis is the Kendall Tou coefficient between the model and the real final ranking

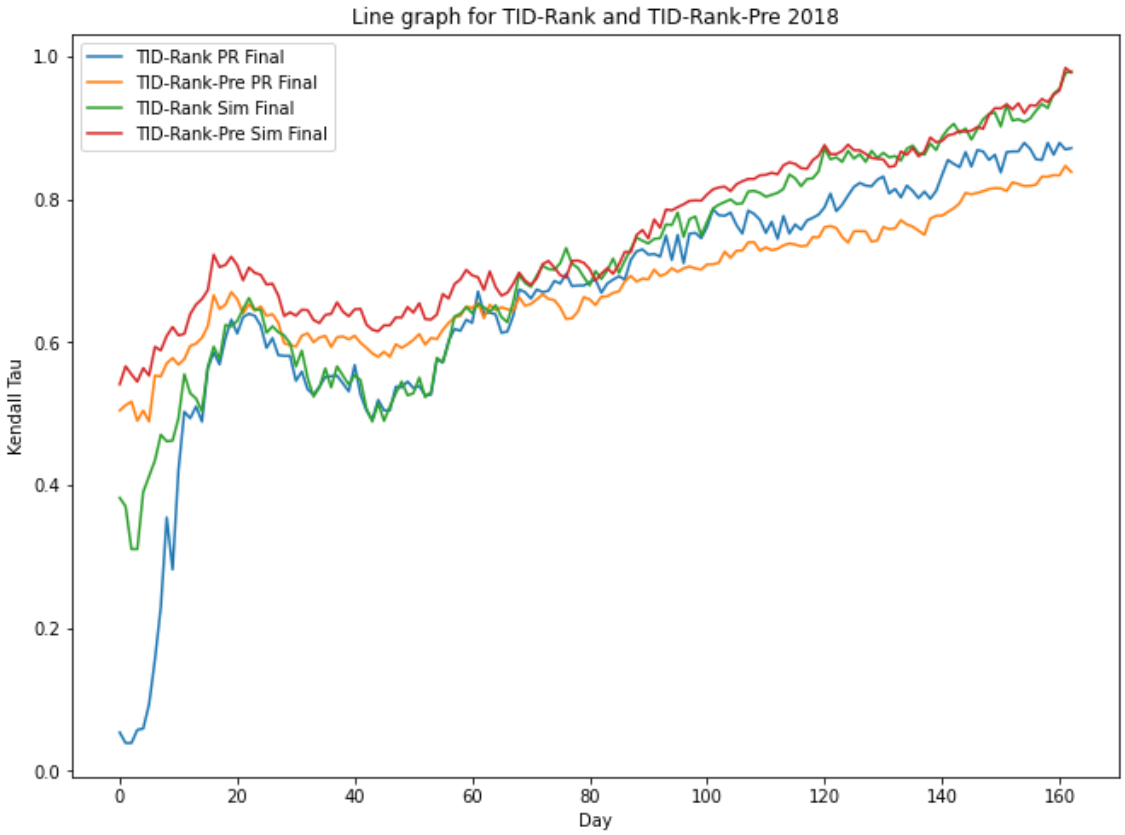

Figure 4.9: Line graph of TID-Rank and TID-Rank-Pre for season 2018
x-axis is the day of the season
y-axis is the Kendall Tou coefficient between the model and the real final ranking

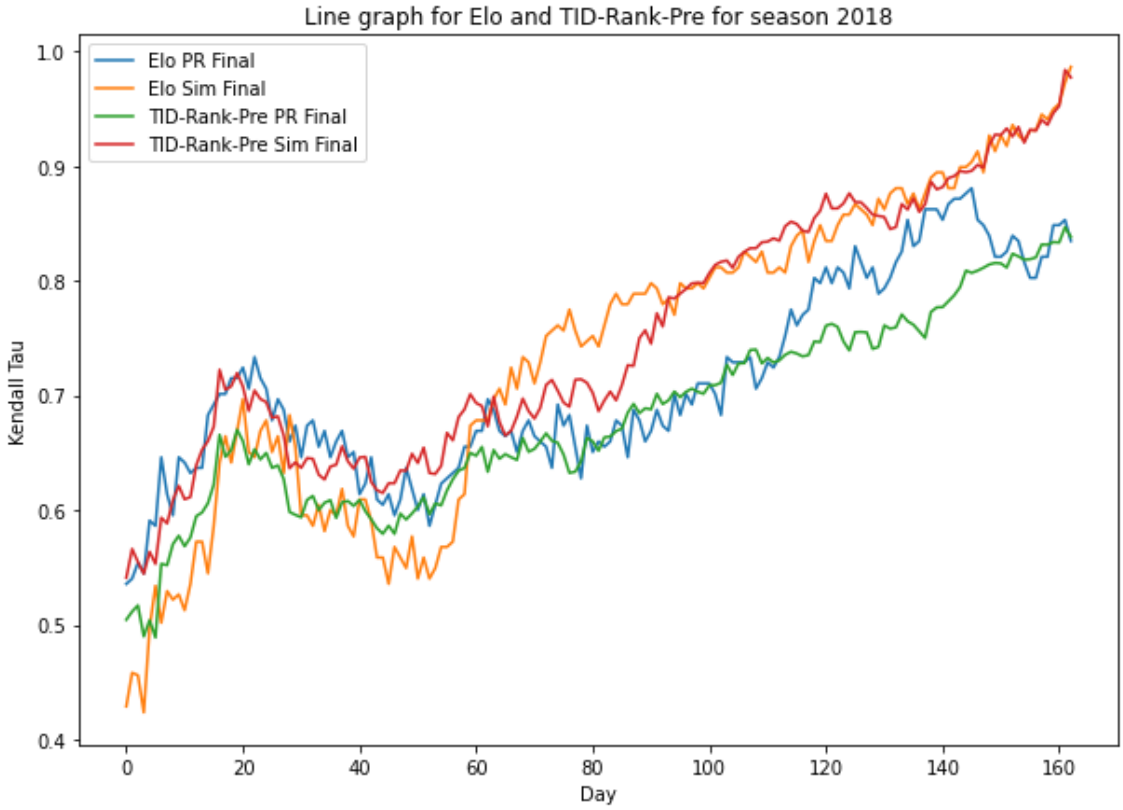

Figure 4.10: Line graph of Elo and TID-Rank-Pre for season 2018
x-axis is the day of the season
y-axis is the Kendall Tou coefficient between the model and the real final ranking

4.3.3 TID-Rank-Pre and ELO for Season 2012-2018

We compared our model against the Elo method models for several seasons. In Figures 4.10 and 4.11, we found out our models have better accuracy on the season outcome in the first half of the season on Sim Final.

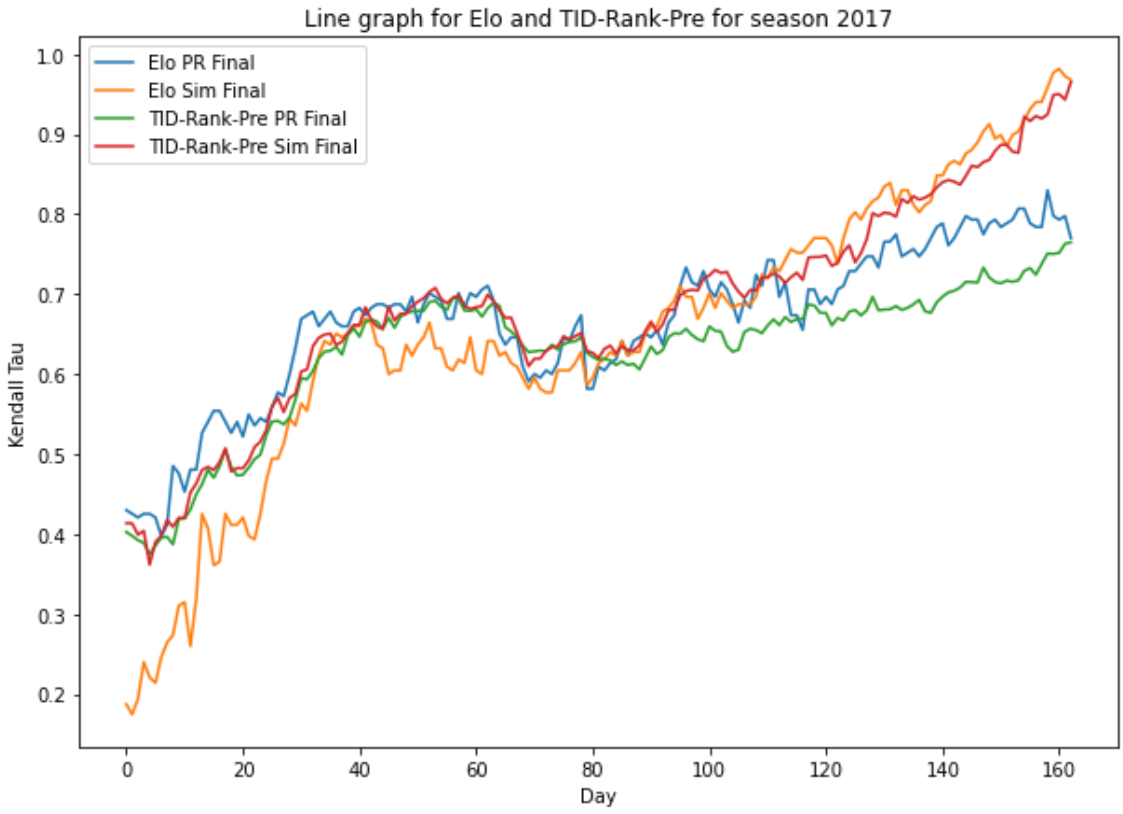

Figure 4.11: Line graph of Elo and TID-Rank-Pre for season 2017
x-axis is the day of the season
y-axis is the Kendall Tou coefficient between the model and the real final ranking

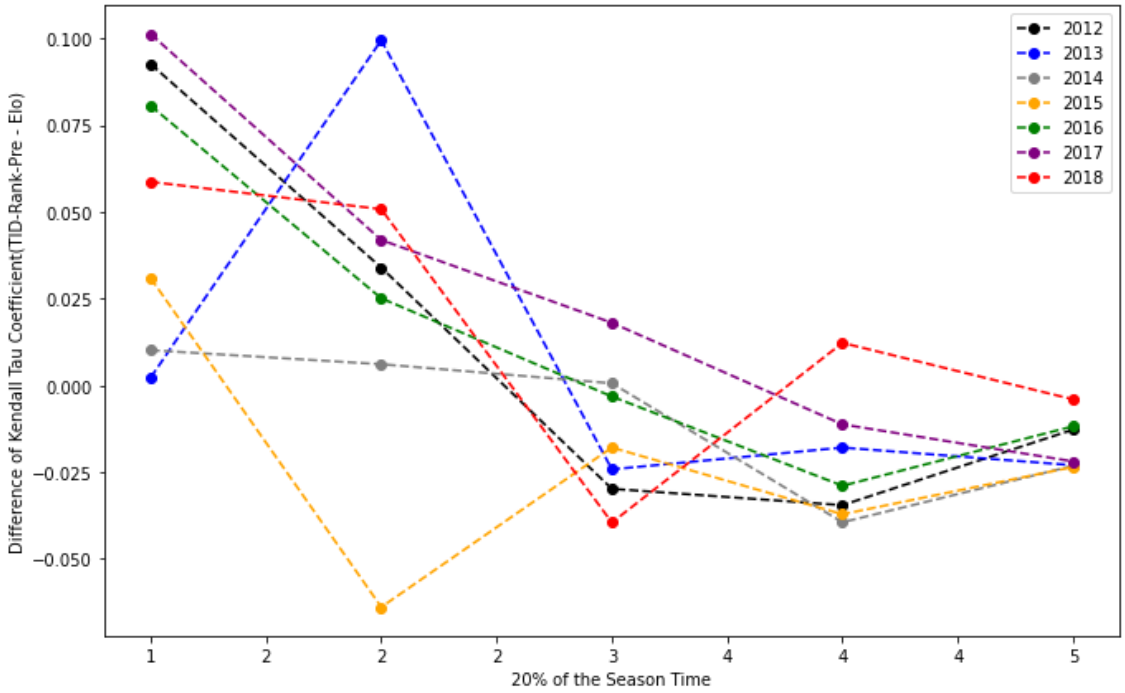

Figure 4.12: Line graph of Differences on Kendall Tau Coefficient of TID-Rank-Pre and Elo for season 2012-2018
x-axis is the day of the season
y-axis is (the Kendall Tou coefficient between TID-Rank-Pre and the real final ranking) - (the Kendall Tou coefficient between Elo and the real final ranking)

Therefore, we split every season into 5 portions by the day and only focus on the difference of Kendall Tau between TID-Rank-Pre and Elo method. In Figure 4.12, we can see that our models make better prediction than the Elo method especially in the first half of the season due to being able to learn from the previous season.

4.3.4 Discussion of the Results

We found out that the curve lines of PR Final and Sim Final do not overlap. The reason is because PR Final does not use accumulated record data, so it is not as accurate in the last quarter of the season. However, we do not know that is our PR Final is actually worse than Sim Final for ranking the strength of all teams at the time because it is impossible to let a team play all the other teams at the same time hypothetically. TID-Rank-Pre model makes better prediction than the Elo method especially in the first half of the season due to being able to learn from the previous season and it only take 1 more dimension to pass in the previous season data which shows the benefit of using deep learning on predicting the ranking.

Chapter 5

Conclusion and Future Work

This chapter will cover our conclusion and what addition research can be done in the future.

5.1 Success

Our individual game outcome *a priori* models differ from some common methods by its non-preset parameters in the models and the accuracy is similar in NBA. Also, we found out that the machine can learn from only know the Team ID and home win advantage, so it might implies that it can build its own ranking system during training. Our ranking method even outperformed Elo method on the first half of the season which shows the benefit of having unfixed parameters. We also learn that for our neural network, team ID and home court advantage are the two most important data input for NN model. So our best model for individual game *a priori* is our TID model because it took the least amount of data and performed among the best models. To our knowledge, there are no NN models which can predict the play-by-play data for basketball at the time we performed our research. This play-by-play model can be an interesting model for people who care about the odds during

the game.

5.2 Limitation

By calculating the sports book odds, we are not positive that we can profit from sports betting market because the margin of the odds is too big to overcome the profit margin of the sports betting company. Furthermore, we did not come up with a good method for the uncertainty, such as injury reports or trades which may effect the outcome, but mainly because there is no sufficient data about those, and we do not think that the NN model can classify them well. The NN models cannot be as effective if there is not enough training data and the uncertainty of sports makes the models hard to catch the sequential trend especially there are only 1230 games in a regular NBA season. Another limitation of the research is any power ranking methods cannot be truly tested since it is not possible to make one team to play against all the other teams at the same time, so it is just a hypothetical theory on ranking, but it can arguably be a better way of ranking the strength of teams.

5.3 Future Work

We did not exhaustively test all network sizes or other optimizer because of the time constraint. In addition, the same type of models we built should be able to apply on other sports tournaments or leagues by tweaking some data entries, but unfortunately we did not have enough time to test it out the robustness of our models on all the sports. In addition, we did not use the individual player data to build our models which may be another interesting topic for sports prediction.

Bibliography

[1] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938, 2018.

[2] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017.

[3] David Aldous. Elo ratings and the sports model: A neglected topic in applied probability? *Statistical science*, 32(4):616–629, 2017.

[4] Eduardo Cabral Balreira, Brian K Miceli, and Thomas Tegtmeyer. An oracle method to predict nfl games. *Journal of Quantitative Analysis in Sports*, 10(2):183–196, 2014.

[5] Olaf RP Bininda-Emonds, Kate E Jones, Samantha A Price, Marcel Cardillo, Richard Grenyer, and Andy Purvis. Garbage in, garbage out. In *Phylogenetic supertrees*, pages 267–280. Springer, 2004.

[6] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. Routledge, 2017.

[7] Rory Bunker and Teo Susnjak. The application of machine learning techniques for predicting results in team sport: a review. *arXiv preprint arXiv:1912.11762*, 2019.

[8] Davide Castelvecchi. Can we open the black box of ai? *Nature News*, 538(7623):20, 2016.

[9] Timothy P Chartier, Erich Kreutzer, Amy N Langville, and Kathryn E Pedings. Sports ranking with nonuniform weighting. *Journal of Quantitative Analysis in Sports*, 7(3), 2011.

[10] Yihang Cheng, Yuanyuan Qiao, and Jie Yang. An improved markov method for prediction of user mobility. In *2016 12th International Conference on Network and Service Management (CNSM)*, pages 394–399. IEEE, 2016.

[11] Dursun Delen, Douglas Cogdell, and Nihat Kasap. A comparative analysis of data mining methods in predicting ncaa bowl outcomes. *International Journal of Forecasting*, 28(2):543–552, 2012.

[12] Lynn E Eberly. Multiple linear regression. *Topics in Biostatistics*, pages 165–187, 2007.

[13] Thomas Viehmann Eli Stevens, Luca Antiga. *Deep Learning with Pytorch*. Manning Publications Co, 2020.

[14] Andrea Giontella, Francesca Maria Sarti, Giovanni Paolo Biggio, Samira Giovannini, Raffaele Cherchi, Maurizio Silvestrelli, and Camillo Pieramati. Elo method and race traits: A new integrated system for sport horse genetic evaluation. *Animals*, 10(7):1145, 2020.

[15] Christina Gough. National basketball association total league revenue from 2001/02 to 2020/21. *Statista*, 2022.

[16] Anjela Yuryevna Govan et al. Ranking theory with application to popular sports. 2008.

[17] George Kyriakides, Kyriacos Talattinis, and Stephanides George. Rating systems vs machine learning on the context of sports. In *Proceedings of the 18th Panhellenic Conference on Informatics*, pages 1–6, 2014.

[18] Amy N Langville and Carl D Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380, 2004.

[19] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[20] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[21] Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[22] OSAA. Osaa rankings frequently asked questions. *Oregon School Activities Association*, 2022.

[23] P. O'Donoghue, D. Ball, J. Eustace, B. McFarlan, and M. Nisotaki. Predictive models of the 2015 rugby world cup: accuracy and application. *International Journal of Computer Science in Sport*, 15(1):37–58, 2016.

[24] S Unnikrishna Pillai, Torsten Suel, and Seunghun Cha. The perron-frobenius theorem: some of its applications. *IEEE Signal Processing Magazine*, 22(2):62–75, 2005.

[25] Grand View Research. Sports betting market size, share and trends analysis report 2021 - 2028. *Market Analysis Report*, 2021.

[26] Baback Vaziri, Shaunak Dabadghao, Yuehwern Yih, and Thomas L Morin. Properties of sports ranking methods. *Journal of the Operational Research Society*, 69(5):776–787, 2018.

[27] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*, 2017.

Appendix A

Additional Figures

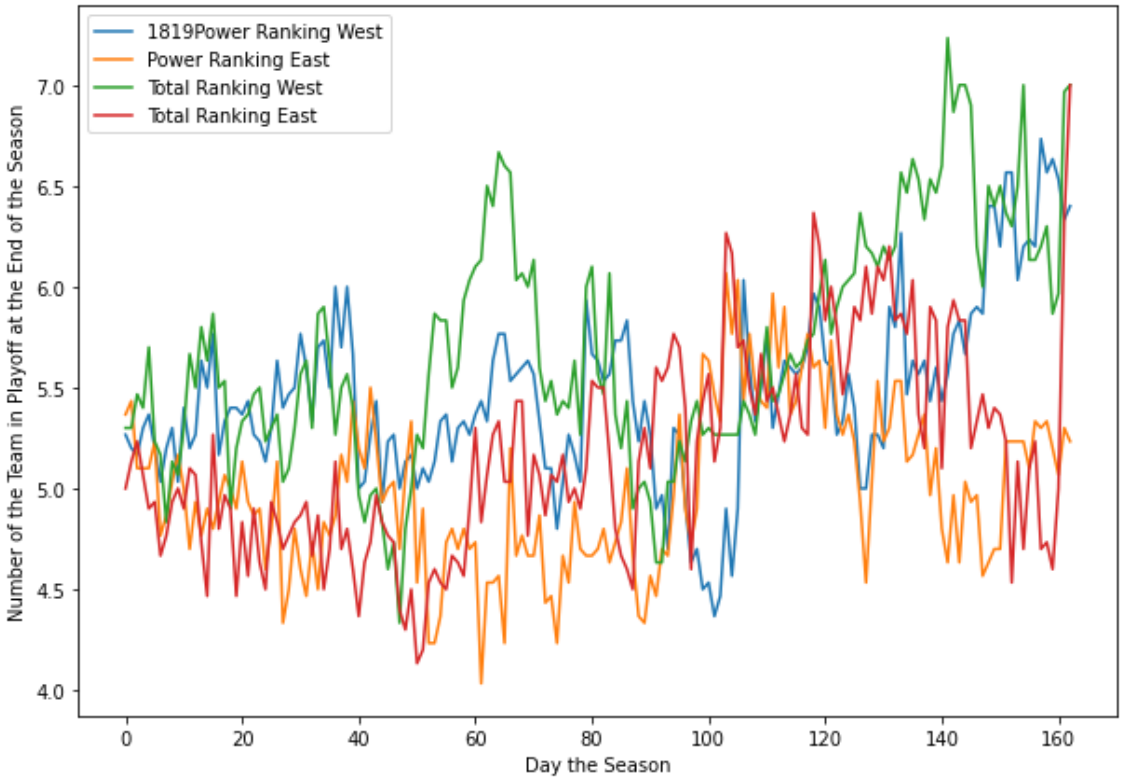

Figure A.1: Using TID-Rank-Pre model to predict which team can make it to the playoff by using our Sim-Final and PR-Final methods and splitting the teams into East and West

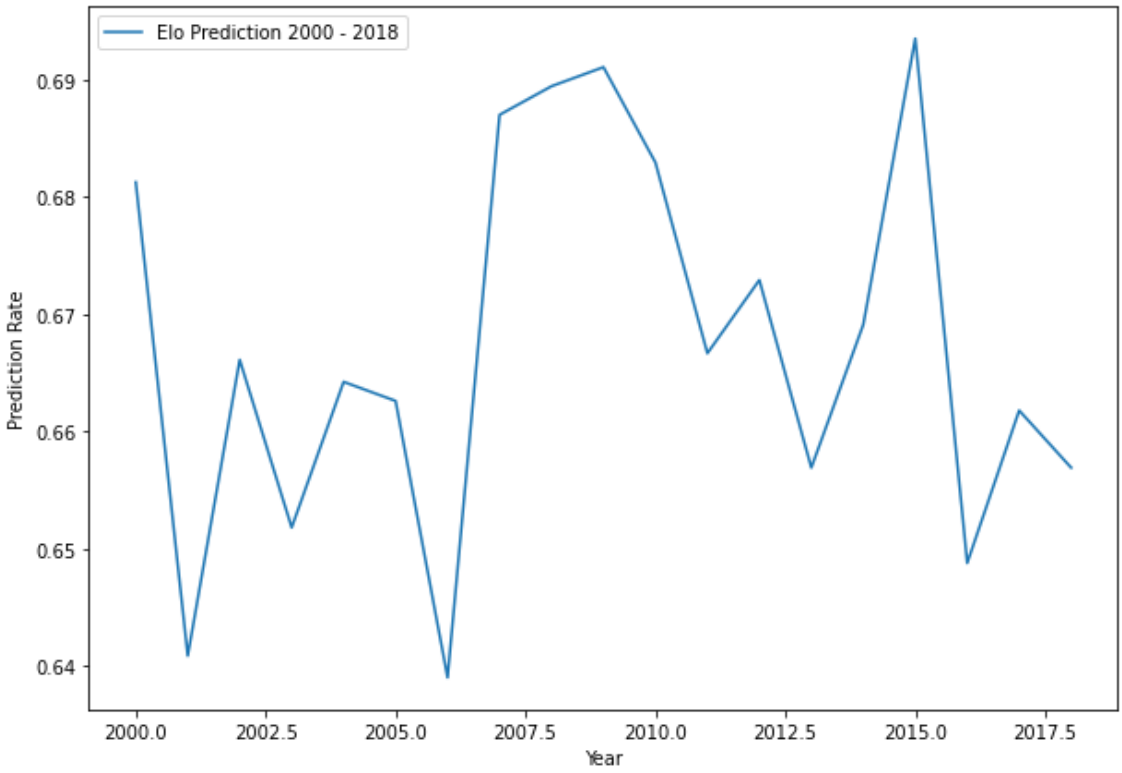

Figure A.2: The line plot shows the prediction accuracy of elo throughout season 2000 to 2018