Predicting the Outcome of Games based on Graph Neural Networks

[1]*Yen-Tsang Wu*, [1,*] *Jenq-Haur Wang, and* [2]*Ning Chien*

[1] Department of Computer Science and Information Engineering,
National Taipei University of Technology, Taipei, Taiwan,
[*]E-mail: buddyswu@gmail.com; jhwang@ntut.edu.tw
[2] Chunghwa Telecom Laboratories Advanced Technology Laboratory, Taiwan
E-mail: nickchien@cht.com.tw

ABSTRACT

In recent years, most studies on predicting the outcome of games faced two main issues. Firstly, player statistics recorded after the game are used to predict the result. Secondly, the analysis and prediction are based on the team's average performance. Furthermore, these studies primarily employ conventional statistical methods for prediction, without considering the correlations between the data, leading to poor performance in predicting outcomes. This paper proposes a method based on a time series model and graph neural network to predict the outcomes of playoff games. Firstly, a graph neural model is used where each player is a node, and the predicted player performance from the time series model is used as the node features. The positional relationships of players in a team are the edges. Secondly, a graph neural network model is trained for prediction. From the experimental results on the National Basketball Association (NBA) data for the 2020-2021 season, the prediction accuracy of the proposed method reached 76.9%. This shows the effectiveness of the proposed method for predicting the outcome of games.

Keywords: *Player performance prediction, NBA game outcome prediction, graph neural network, machine learning.*

1. INTRODUCTION

With the booming development of the sports economy, the NBA has become one of the fastest-growing industries worldwide. Existing game outcome predictions mainly rely on post-game data, neglecting the importance of pre-game data for prediction. Furthermore, most current studies use average team data for prediction, lacking consideration of individual player performance. Therefore, they fail to address two major issues in predicting team outcomes: (1) insufficient pre-game player performance prediction and (2) lack of attention to interactions between players. To address these problems, we propose using Gated Recurrent Unit (GRU) and Graph Neural Networks (GNN) from deep learning to improve prediction accuracy.

Our contributions are as follows: Firstly, for data prediction, we use time series models to predict player performance before the game, instead of traditional mathematical calculations, providing more realistic game outcome predictions. Secondly, this study utilizes the performance data of all participating players and applies graph neural networks for game prediction for the first time, achieving an accuracy of 76.9%, surpassing existing methods.

2. RELATED WORK

Due to the extensive amount of statistical data available for NBA games, there has been significant related research. In 2015, Greene [1] utilized a comprehensive data model to estimate the NBA draft rankings of college rookies. This study highlighted the use of detailed statistical analysis and data modeling to enhance the understanding of player potential in the NBA draft. In 2019, Hu et al. [2] used neural networks to predict the MVP of the NBA season. In 2021, Sarlis et al. [3] applied deep learning to evaluate how injuries affect individual player performance and overall team results. In 2024, Farghaly et al. [4] used various machine learning methods to explore the likelihood of lower body muscle injuries in NBA players and analyzed various factors affecting injury risk. Numerous studies have used spatial and temporal data, deep learning, ensemble learning, and statistical methods to predict playoff outcomes, player injuries, and draft rankings. This study focuses on predicting NBA game outcomes, and the related literature can be categorized into three main areas: basketball game outcome prediction, feature selection, and the application of GNN models in sports events.

2.1 Basketball Game Outcome Prediction

Game outcome prediction can be considered a binary classification task. Hu et al. [5] used numerical analysis to study the differences in home and away wins, calculating the outcome of the 1996-1997 championships using weighted likelihood. Miljković et al. [6] divided game data into home and away games, using Naïve Bayes and multivariate linear regression as classification models, achieving an accuracy of 67%. Cao [7] used data from the 2005-06 to 2009-10 seasons for training and predicted the 2010-11 season, using models such as Naïve Bayes Classification, SVM, and Logistic Regression, reaching an accuracy of 69.67%.

Pai et al. [8] proposed an HSVMDT framework, which combines SVM and Decision Tree models, achieving an accuracy of 85.2%. Jain et al. [9] proposed a Hybrid Fuzzy-SVM (HFSVM) to reduce noise in the data, which negatively affects SVM performance, and used CFS [10] for feature selection, achieving an accuracy of 88.26%. Horvat et al. [11] experimented with multiple seasons worth of data using methods such as Decision Tree, K-NN algorithm, and Random Forest, achieving an accuracy of 60.8%. Osken et al. [12] used K-Means and C-Means clustering to identify player types, training prediction models based on cluster member capabilities. Their method achieved a prediction accuracy of 76% over five NBA seasons. In Wang's study [13], feature engineering was used to improve the model's prediction accuracy by analyzing and selecting key metrics. The study found that field goal percentage (FG%), three-point percentage, and steals were key statistics for predicting game outcomes. They used Random Forest and DNN for prediction, achieving an accuracy of 74%. Adam et al. [14] used first-half statistics and SVM for outcome prediction, achieving an accuracy of about 66.67%.

2.2 GNNs Methodology Used in Sports

Graph Neural Networks (GNNs) are neural networks designed to handle graph-structured data, leveraging the relationships and interdependencies between nodes and edges for information propagation and learning. Xenopoulos [15] used GNNs to predict outcomes of NFL and CSGO games, reducing loss by 9% and 20%, respectively. Zhao et al. [16] combined GCN with RF algorithms to enhance prediction accuracy, achieving an accuracy of approximately 71.54% for NBA predictions. Luo and Krishnamurthy [17] proposed a deep learning method called GATv2-TCN, exploring the use of GAT and Temporal Convolutional Networks for predicting sports performance.

2.3 Prediction Based on Feature Selection

Due to the numerous attributes of NBA data, feature selection is crucial for prediction performance. Thabtah et al. [18] added home and away factors to traditional data, using Multiple Regression [19], CFS, and the RIPPER algorithm [20] for feature selection. The best-performing model was a combination of RIPPER and Bayes, achieving an accuracy of 83%. They also found that the feature "DRB" was common in all five selected groups, indicating its importance in NBA game outcome prediction.

3. METHODOLOGY

The methodology proposed in this study comprises of two components. The first component involves predicting player performance using a time series model. Since we cannot obtain player performance before the game starts, we need to predict the performance of the players based on their past performances. The second component utilizes a graph neural network (GNN) to predict the outcome of the playoffs. Prior to making predictions, we will use feature selection to identify the key player performance features that most affect the game outcome.

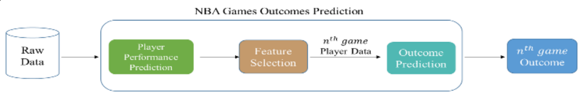
NBA Games Outcomes Prediction
Raw Data
Player Performance Prediction
Feature Selection
n^{th} game Player Data
Outcome Prediction
n^{th} game Outcome

Figure 1. System process flow

The system process is illustrated in Figure 1. This study comprises three main steps: player performance prediction, feature selection, and game outcome prediction. First, we standardize the features and employ a GRU model [21] to predict player performance, using data from multiple previous games as input to calculate the output of a single feature and predict the player's performance for the next game. Mean squared error (MSE) is employed to evaluate the time series model, which is the sum of the squares of the differences between predicted and actual values. Advanced data features, such as field goal percentage (FG%) = field goals made (FG) / field goals attempted (FGA), are calculated from other data and are not predicted by the time series model.

3.1 Player Performance Prediction Model

Before the game begins, it is impossible to determine how players will perform. Thus, predicting player performance is a key factor. Observing past data, we find that player performance tends to maintain a certain trend over time. Therefore, we treat player performance prediction as a time series forecasting task. The model's input is the player's performance data, which must be transformed into the format (n, m, k) to train the GRU. Here, n represents the number of games in the input batch, m is the size of the sliding window, and k is the player of performance features. The transformed data is then fed into the time series model to train the GRU. After passing through a linear layer, the GRU's output vector is converted into real values to predict player performance for the subsequent (n+1) game. Since there are a total of 21 features, the prediction process is repeated 21 times to complete the forecast of player performance. The proposed architecture is shown in Figure 2.

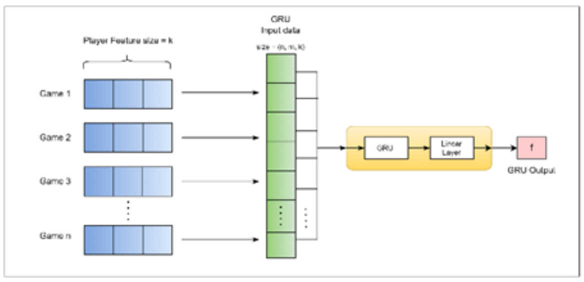

Figure 2. Player Performance Prediction Architecture

In each iteration, only one feature is predicted. The sliding window method utilizes data from the preceding *n* games to predict player performance in game *n+1*. Figure 3 provides an example of using the first three games' player performances to predict the performance for the fourth game, where g_i represents the *i-th* game and $i \in [1, n]$.

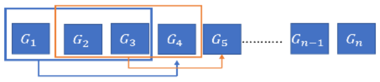

Figure 3. Sliding Window Illustration

3.2 Feature Selection

The publicly available player data provided by the NBA is categorized into traditional and advanced data. Advanced data is calculated from traditional data, but the NBA does not disclose the calculation methods. Therefore, the time series model cannot predict advanced data, and this study utilizes traditional data for experiments such as assist rate (AST%), total three-pointers (TPA), and offensive rebounds (ORB) etc." After predicting the player performance data, feature selection is conducted to determine which player features are most important for predicting the game's outcome. Support Vector Machine (SVM) is used for this purpose.

3.3 Playoff Outcomes Prediction

In a game, players' contributions to the final outcome vary, and it is crucial to identify the key players and the relationships between them. This paper proposes the **Graph Attention Convolutional Network (GATCN)**, a framework that integrates the GAT [22] and GCN [23] for game outcome prediction, as shown in Figure 4.

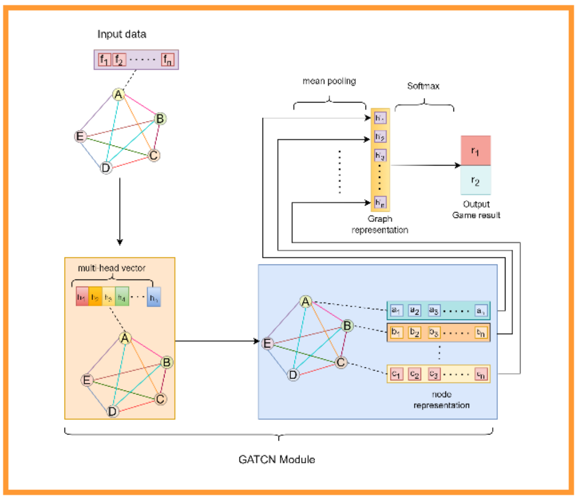

Figure 4. GATCN Architecture

Player performance data is converted into a graph format and used as input for the GATCN. The GAT model computes the one-hop feature vectors, updating node feature vectors through attention coefficients. The GCN aggregates both local and global connections between nodes to generate updated node representations. These representations are subsequently transformed into a graph-level representation using mean pooling, and finally, a softmax function outputs the game's predicted outcome.

3.3.1 Player-to-Player Graph

To accurately assess the influence of each player, we propose a **Player-to-Player Graph** construction method. This method creates a graph based on the players who participated in the game, representing their relationships. Given a graph *G*=(*V*,*E*), where *G* represents a game, *V* is the set of players *p*, expressed as $V = \{p_1, p_2, \dots, p_n\}, n \in \mathbb{R}$, where *n* is the number of players on a team. The edges $E = \{e_1, e_2, \dots, e_n\}, n \in \mathbb{R}$ reflect the relationships between players. Since the NBA does not officially provide information on playing time for starting and bench players, we construct edges based on team relationships, forming a fully connected graph, as shown in Figure 5**.**

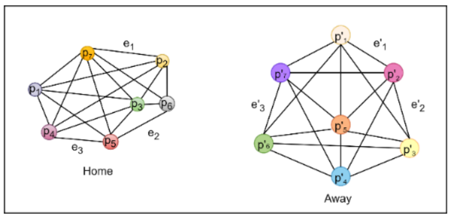

Figure 5. Player-to-Player Graph Illustration

The edges are represented by an adjacency matrix A, where A_{ij} represents the edge between players p_i and p_j, where $i, j \in \mathbb{R}$. Edges are established between players on the same team, as defined in Equation (1). If p_i and p_j are on the same team, the value of the adjacency matrix A_{ij} is 1; if they are not on the same team, A_{ij} is 0, forming an undirected graph. The node features represent each the players' performance in the match.

$$f(\mathrm{x}) = \begin{cases} 1, & if\ (p_i, p_j) \in E\ and\ i \neq j \\ 0, & \text{others} \end{cases} \tag{1}$$

3.3.2Graph Attention Convolution Network (GATCN)

The proposed GATCN integrates techniques from both GAT and GCN. During the propagation process, GATCN utilizes a self-attention mechanism to compute attention coefficients between each node and its neighbors, thereby determining the importance of each node, as defined in Equation (2).

$$\alpha_{ij} = \frac{\exp\left(LeakyReLU\left(\vec{a}^T[\mathbf{W}\vec{h}_i \parallel \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in N_i}\exp\left(LeakyReLU\left(\vec{a}^T[\mathbf{W}\vec{h}_i \parallel \mathbf{W}\vec{h}_k]\right)\right)} \tag{2}$$

Here, α_{ij} denotes the attention coefficient between nodes (i, j), where $i, j \in \mathbb{R}$. The LeakyReLU function [24] is applied as an activation function to adjust the weights. We employ multi-head self-attention, enabling each head to focus on both local and global information. To integrate the feature vectors generated by each head, we concatenate them as described in Equation (3).

$$h_i' = \parallel_{k=1}^{K} \sigma\left(\sum_{j\in N_i} \alpha_{ij}^k W^k h_j\right) \tag{3}$$

Here, $\parallel$ represents concatenation, K is the number of heads, and σ represents the sigmoid activation function. α_{ij}^k denotes the attention coefficient between nodes i and j in the k-th head, $W^k$is the linear transformation weight matrix for the k-th head, and h_i' is the new attention coefficient after concatenating the attention outputs from each head, representing the weight of each player in the game. Since the graph has local connectivity, we employ convolution to propagate features. At this stage, the features are already computed by GAT, so they include the 1-hop neighbors' attention. The GCN subsequently aggregates the feature information from all neighbors and the node itself. After pooling the updated node feature vectors, we transform them into graph-level feature vectors using **mean pooling**, defined in Equation (4).

$$x_i = \frac{1}{N_i}\sum_{n=1}^{N_i} h_i' \tag{4}$$

Here, N_i represents the number of neighboring nodes, and x_i is the graph embedding representing all player information in a game. After passing through the softmax function, we obtain the final output, which is a two-dimensional vector representing the probabilities of winning or losing the game. During training, we apply the cross-entropy loss function to prevent overfitting by calculating the loss H_i at each training iteration, as defined in Equation (5).

$$H_i = \sum_{c=i}^{C}\sum_{i=1}^{n} -y_c \log_2(r_{c,i}) \tag{5}$$

Here, C represents the number of categories (win/loss), n is the total number of samples, y_c is the one-hot encoded label for the i-th data, and $r_{c,i}$ denotes the predicted probability for class c for the i-th sample.

4. EXPERIMENTS

4.1 NBA Dataset

The data used in this study was collected through web scraping from www.basketball-reference.com for the 2020-2021 NBA season. Due to the impact of COVID-19, each team played only 72 regular-season games, totaling 1,080 games. The playoffs comprised 91 games. As each game involves two teams, with one win and one loss, this results in a total of 2,160 data points. The dataset includes traditional player statistics, advanced statistics, team averages, and game outcomes (labels). To avoid redundant predictions, we focus on home team player performance as the prediction target, with an 8:2 split between training and validation sets.

4.2 Evaluation Metrics

We evaluate the model's performance using accuracy, precision, recall, and F1-score to assess its effectiveness and usability.

4.3 Player Parameters

Since our model predicts game outcomes based on player performance, the number of players participating in the game is a crucial factor. We analyzed data from 681 playoff games spanning the 2016–2019 seasons. According to NBA statistics, we found that bench players typically play for 15–20 minutes, with the most common number of players per game being 7, observed in 280 games, followed by 8 players in 203 games. Therefore, we set the number of players in the model to either 7 or 8 for our experiments.

4.4 Model Comparison

To verify the effectiveness of the proposed model, we compared it with the following models:

- **Baseline**: This model combines methods from two studies. First, it applies Jones' [25] 3-game-average method to predict player performance, followed by XGBOOST for feature selection. Next, Thabtah et al.'s [18] Artificial Neural Network (ANN) model is used to predict game outcomes.
- **GAT**: Based on the framework proposed by Velickovic et al. [22], this model incorporates attention mechanisms into graph neural networks. It focuses on a single node and its 1-hop neighbors, calculating attention coefficients to update node feature vectors.
- **GCN**: Proposed by Yao et al. [23], this model employs graph convolutional networks for graph classification. It was originally applied to text classification. The architecture includes two convolution layers and two fully connected layers. The GCN produces 64-dimensional vectors, which are then reduced to 2-dimensional vectors via softmax for binary classification.
- **DGCNN**: An improved GCN model proposed by Zhang et al. [26], DGCNN consists of four GCN layers. It employs sort pooling to rank node roles in

the graph based on their structural positions, followed by a traditional 1-D convolution. The output is then processed through a linear layer for classification.

4.5 Experimental Results

Table 1 shows the experimental results comparing the proposed method with other models:

Table 1. Summary of Experimental Results

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
Baseline	0.593	0.886	0.602	0.717
SVM + GAT	0.736	0.754	0.784	0.769
SVM + GCN	0.736	0.792	0.763	0.777
SVM + DGCNN	0.747	0.773	0.788	0.781
GATCN	0.652	0.769	0.666	0.714
SVM + GATCN	0.769	0.811	0.796	0.803

The experimental results indicate that our proposed model, which employs a sliding window of 3 games as input to the GRU for player performance prediction and constructs graphs with 8 players, achieves the highest performance in accuracy (0.769), recall (0.796), and F1-score (0.803). The findings demonstrate that the GATCN model, after feature selection using SVM, is more effective at predicting game outcomes than other models. By aggregating each node's one-hop neighbors through GAT and updating node features with weighted averaging, GATCN further incorporates the Laplacian matrix operation via GCN to aggregate all neighboring features, surpassing models that utilize only GAT or GCN. Specifically, GATCN outperforms both GAT and GCN models in accuracy by 3.3%. This performance demonstrates that GATCN effectively combines the strengths of GAT and GCN. The DGCNN model shows an F1-score of 0.781, utilizing multi-layer GCNs and sort pooling to compute the relationships between nodes and neighbors. However, without the help of attention coefficients, its performance is still lower than GATCN.

5. ANALYSIS AND DISCUSSION

5.1 Analysis of Feature Selection

The selection of player data for features is a critical factor in the model's performance. We applied SVM to evaluate the importance of each feature, as illustrated in Figure 6. Features with values close to 1 are associated with winning, while features close to -1 are associated with losing. We selected [TRB, STL, PTS, FG%, BLK, FT%, +/-, 3P%, ORB, DRB, FGA, TOV, FG, FTA, 3P, FT] as game features, with TRB having the highest value, indicating a strong correlation with winning, and ORB has the lowest value, indicating a strong correlation with losing. Features such as [MP, AST, PF, 3PA] were excluded as their values were close to 0, suggesting minimal relevance to game outcomes.

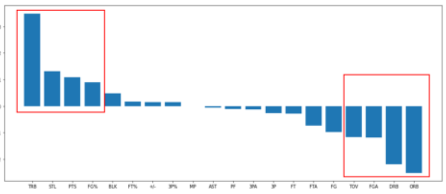

Figure 6. SVM Feature Importance Scores

In previous studies, common feature selection methods include the chi-square test, Random Forest (RF), and XGBoost. Since these methods calculate feature scores, this study compares feature selection methods by selecting features with scores higher than the mean [27]. In addition to individual player data, the number of players used to form edges is also a key parameter. As noted in the previous section, the most commonly used number of players per game is 7 to 8. Accordingly, we conducted experiments using 7 and 8 players. To determine the best features, we applied the four feature selection methods mentioned above to player data generated with various sliding window sizes and tested them with different numbers of players using the GATCN model. The sliding windows are represented by the number of games. The experimental results are as follows.

Table 2. Experimental Results of Feature Engineering

Game	**Players**	**Method**	**Accuracy**	**Precision**	**Recall**
2	8	Baseline	0.652	0.692	0.692
2	8	SVM	0.681	0.717	0.730
3	**8**	**SVM**	**0.769**	**0.811**	**0.796**
3	8	XGBOOST	0.758	0.811	0.781
4	8	SVM	0.637	0.679	0.692
4	8	XGBOOST	0.626	0.660	0.686

From Table 2, it is evident that constructing graphs with 8 players and using 3 games as GRU input yields superior accuracy, precision, recall, and F1-score compared to using 2-game or 4-game as input. Notably, performance with 4-game input is the lowest, slightly underperforms the 2-game input in both XGBoost and RF models. This suggests that player performance does not exhibit long-term patterns conducive to GRU learning, with 3 games proving to be the most optimal input. Using SVM for feature selection achieved the highest accuracy, recall, and F1-score values of 0.769, 0.796, and 0.803, respectively. However, in terms of precision, the 2-game and 4-game inputs with RF feature selection achieved a precision of 1. Closer inspection reveals that this outcome results from all predictions favoring game victories, yielding the highest precision but a significantly lower accuracy of 0.582.

Regarding feature selection methods, SVM consistently provided the best performance, followed by XGBoost. The features identified by these two methods were relatively similar, with 8 common features: TRB, STL, PTS, FG%, BLK, FT%, +/-, and 3P%. As a result, their performance across all four-evaluation metrics was relatively close. XGBoost determines feature importance based on gain, which reduces model entropy and enhances crucial information, leading to more precise feature importance. In contrast, Random Forest calculates importance by aggregating the Gini index across multiple trees. Variations in tree splitting due to differences in the Gini index contributed to inconsistent results in this experiment. Lastly, the chi-square test, which is more suited to categorical data analysis, struggled to identify key features in this dataset due to feature similarity between winning and losing games, as calculating expected values did not reveal distinguishing characteristics.

5.2 Analysis of Time Series Model

We compare the performance of two well-known time series models, GRU and LSTM. Both models are trained with a sliding window size of 3 games, 8 players, and 50 epochs. The optimizer used is Adam [28], and the loss function is MSE. After predicting player performance with each model, we use GATCN to predict the game outcomes. According to the experimental results, we can see that GRU achieves convergence across all features by the 15th epoch, while LSTM's feature converges around the 20th epoch. Both models employ SVM-selected features for prediction. The GRU model outperforms LSTM across all evaluation metrics, demonstrating significant improvement by replacing the input and forget gate with a single update gate. During calculations, GRU integrates hidden layer information into unit state calculations, allowing for more comprehensive information transfer. In contrast, LSTM separates the calculations of the hidden layer and unit state, transferring only partial information, which leads to differences in accuracy and performance. Furthermore, GRU’s simpler structure reduces training time by 2.5%.

5.3 Analysis of GATCN

To assess the effectiveness of our proposed GATCN model, we compared it with the models proposed by Jain and Kaur [9] and Zhao et al. [16]. Since Jain et al.'s HFSVM model uses the 2015-2016 NBA season for experiments, we also applied our proposed model to the 2015-2016 season for consistency in comparison. The experimental results showed that GATCN achieved an accuracy of 0.9302, while HFSVM achieved 0.8826, and Zhao's model had a maximum accuracy of 0.707. The results indicate that the proposed GATCN improves accuracy by 4.8% over Jain et al.'s model and by 22% over Zhao's model. This demonstrates that for predicting NBA game outcomes, player performance data provides more precise information than team-level data alone.

6. CONCLUSION AND FUTURE WORK

In this study, we proposed a method for predicting NBA playoff outcomes. First, we used GRU to predict future player performance, treating each player as a node and connecting all nodes to form a graph. Feature selection was conducted using SVM, and the final prediction was made with the proposed Graph Attention Convolutional Network (GATCN). Compared to solely predicting based on player performance, incorporating structural relationships between players resulted in more accurate model predictions. The experimental results show that the proposed GATCN model outperforming other state-of-the-art methods.

In the future, we aim to enhance the construction of player-to-player graphs by considering specific features to identify which player combinations most influence game outcomes. This approach could be applied not only to game outcome prediction but also to other areas such as team formation and player trades. Regarding model improvements, since the attention mechanism in GAT aggregates only 1-hop neighbors, we plan to refine the attention mechanism to capture more extensive interactions and further improve model accuracy. These are some potential directions for advancing the model.

REFERENCES

1. Greene, A.C., *The Success of NBA Draft Picks: Can College Careers Predict NBA Winners?* 2015.
2. Hu, J., H. Zhang, and J. Qiu. *Prediction of MVP attribution in NBA regular match based on BP neural network model*. in *Proceedings of the 2019 international conference on artificial intelligence and advanced manufacturing*. 2019.
3. Sarlis, V., et al., *A data science approach analysing the impact of injuries on basketball player and team performance.* 2021. **99**: p. 101750.
4. Farghaly, O. and P. Deshpande. *Leveraging Machine Learning to Predict National Basketball Association Player Injuries*. in *2024 IEEE International Workshop on Sport, Technology and Research (STAR)*. 2024. IEEE.
5. Hu, F. and J.V.J.L.N.-M.S. Zidek, *Forecasting NBA basketball playoff outcomes using the weighted likelihood.* 2004: p. 385-395.
6. Miljković, D., et al. *The use of data mining for basketball matches outcomes prediction*. in *IEEE 8th international symposium on intelligent systems and informatics*. 2010. IEEE.
7. Cao, C., *Sports data mining technology used in basketball outcome prediction.* 2012.
8. Pai, P.-F., et al., *Analyzing basketball games by a support vector machines with decision tree model.* 2017. **28**: p. 4159-4167.
9. Jain, S. and H. Kaur. *Machine learning approaches to predict basketball game outcome*. in *2017 3rd international conference on advances in computing, communication & automation (ICACCA)(Fall)*. 2017. IEEE.
10. Hall, M.A., *Correlation-based feature selection for machine learning*. 1999, The University of Waikato.
11. Horvat, T., L. Havaš, and D.J.S. Srpak, *The impact of selecting a validation method in machine learning on predicting basketball game outcomes.* 2020. **12**(3): p. 431.
12. Osken, C. and C.J.H. Onay, *Predicting the winning team in basketball: A novel approach.* 2022. **8**(12).
13. Wang, J. *Predictive Analysis of NBA Game Outcomes through Machine Learning*. in *Proceedings of the 6th International Conference on Machine Learning and Machine Intelligence*. 2023.
14. Adam, C., P. Pantatosakis, and M. Tsagris, *On predicting an NBA game outcome from half-time statistics.* 2024.
15. Xenopoulos, P. and C. Silva. *Graph neural networks to predict sports outcomes*. in *2021 IEEE International Conference on Big Data (Big Data)*. 2021. IEEE.

16. Zhao, K., C. Du, and G.J.E. Tan, *Enhancing basketball game outcome prediction through fused graph convolutional networks and random forest algorithm.* 2023. **25**(5): p. 765.
17. Luo, R. and V.J.a.p.a. Krishnamurthy, *Who You Play Affects How You Play: Predicting Sports Performance Using Graph Attention Networks With Temporal Convolution.* 2023.
18. Thabtah, F., L. Zhang, and N.J.A.o.D.S. Abdelhamid, *NBA game result prediction using feature analysis and machine learning.* 2019. **6**(1): p. 103-116.
19. Berger, D.E.J.U.C.G.U., *Introduction to multiple regression.* 2003.
20. Cohen, W.W., *Fast effective rule induction*, in *Machine learning proceedings 1995*. 1995, Elsevier. p. 115-123.
21. Dey, R. and F.M. Salem. *Gate-variants of gated recurrent unit (GRU) neural networks*. in *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*. 2017. IEEE.
22. Veličković, P., et al., *Graph attention networks.* 2017.
23. Yao, L., C. Mao, and Y. Luo. *Graph convolutional networks for text classification*. in *Proceedings of the AAAI conference on artificial intelligence*. 2019.
24. Xu, B., *Empirical evaluation of rectified activations in convolutional network.* arXiv preprint arXiv:1505.00853, 2015.
25. Jones, E.S., *Predicting outcomes of NBA basketball games*. 2016, North Dakota State University.
26. Zhang, M., et al. *An end-to-end deep learning architecture for graph classification*. in *Proceedings of the AAAI conference on artificial intelligence*. 2018.
27. Chen, W.-J., et al., *Hybrid basketball game outcome prediction model by integrating data mining methods for the national basketball association.* 2021. **23**(4): p. 477.
28. Diederik, P.K.J., *Adam: A method for stochastic optimization.* 2014.