Victory in Graph: Battle game result prediction using heterogeneous graph

Dayeon Jeong
School of BBE, KAIST
Daejeon, South Korea
doremikong@kaist.ac.kr

Yunbyeong Chae
School of BBE, KAIST
Daejeon, South Korea
beong0717@kaist.ac.kr

ABSTRACT

Who will be the RULER of the battlefield? In the context of modern warfare, we aim to leverage data analysis, data mining, and machine learning to predict outcomes, bridging the gap between virtual gaming and real military applications, much like how military intelligence relies on predictive analytics for assessing enemy movements and forecasting battlefield conditions. The goal of this project is to predict the outcome of the battle game using given data. There are two types of games: one-versus-one and multiple players. We plan to use three methods to predict game results: Heterogeneous graph neural network, average score, and PageRank. The main idea is to construct a heterogeneous graph with two node types, which are game-player pair node and player node. Then we learn the graph with GraphSAGE algorithm to predict the result of each game. In addition, we will use PageRank and average score methods to represent player node features. By predicting the outcome of the game using graph structures and machine learning, we obtained better prediction results than random methods and methods that simply considered the average score.

1 INTRODUCTION

In the realm of modern warfare, prediction is crucial, as military strategists analyze data to gain insights and make informed decisions. We aim to bridge virtual gaming and real-world military applications, using data mining and machine learning to predict outcomes in virtual battles. Our goal is to answer the question: "Can we accurately forecast the results of virtual battles, similar to how military intelligence predicts enemy movements and battlefield conditions?"

There are two types of virtual battles. One is a one-versus-one match where the winner is determined, and the other is where multiple players participate and scores are given to each player.

An overview of predicting the outcome of both types of battles is as follows:

- GIVEN: player IDs of each game, 2 IDs for one-versus-one matches, 2 to 8 IDs for multiple players matches
- FIND: a battle game result prediction function
- to MAXIMIZE: the accuracy in predicting game results

To address the problems, we propose several methods: Heterogeneous graph neural network(GraphSAGE), PageRank, and average scores of players.

- Heterogeneous graph neural network(GraphSAGE) - The heterogeneous graph consists of 'game-player pair' nodes and 'player' nodes. The information given in the constructed graph is learned using GraphSAGE algorithm to predict the outcome of the matches.
- Average score - The average score takes into account the win probability or the score that a particular player received in previous matches. This is used for both winner prediction and score prediction, and like PageRank, it is used as a node feature in heterogeneous graphs.
- PageRank - PageRank is calculated by creating a directed weighted graph from training data for score prediction. A high PageRank value means that the node has high importance in the directed weighted graph and is used for a node feature in heterogeneous graphs.

The contributions of this project are that it constructed a heterogeneous graph and used graph neural network with deeply analyzed features to develop a prediction model with much higher accuracy than random methods or simple average score prediction methods.

2 PROBLEM STATEMENT

There are two types of tasks. One is a one-versus-one match where the winner is determined, and the other is where multiple players participate and scores are given to each player.

The goal of the first task *Winner Prediction* is to predict the outcome $\mathbf{z} \in \{\mathbf{x_1 win}, \mathbf{x_2 win}, \mathbf{draw}\}$ of the given one-versus-one game $\mathbf{y} \in \mathcal{Y}$ where $\mathbf{x_1}, \mathbf{x_2} \in \mathcal{X}$ are the players participated in $\mathbf{y}$. $\mathcal{X}$ and $\mathcal{Y}$ are the set of players and games respectively.

The goal of the second task *Score Prediction* is to predict each player's score $\mathbf{z} \in [0, 1, ..., 9]$ when multiple players $\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_8} \in \mathcal{X}$ participate in one game $\mathbf{y} \in \mathcal{Y}$. $\mathcal{X}$ and $\mathcal{Y}$ are the set of players and games respectively.

3 EXPLORATORY DATA ANALYSIS

Before constructing and learning a heterogeneous graph neural network, we analyzed the given data to determine features to be used as features of the graph. We tried to use PageRank and average scores as features, and therefore investigated how they actually correlated to the outcomes we want to predict.

First, we looked at the relationship between the results of the two types of games. Interestingly, a comparison between the average winning probability of a player participating in a one-to-one game and the average score that player recorded in a multiple player game showed a positive correlation.(Figure 1) This means that people who are good at one to one games also record relatively high scores in multiple games, and these two characteristics are later used as node features of heterogeneous graphs.

Second, in order to understand how PageRank is related to actual scores, we used the PageRank values calculated from the training data to find out the distribution of the scores of the player with

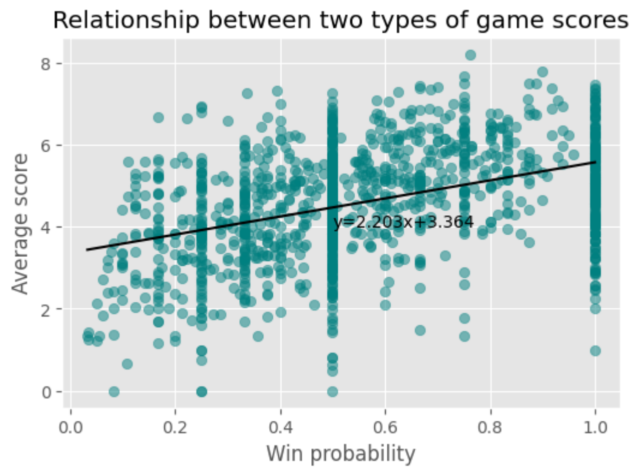

Figure 1: Correlation between two types of game scores

the highest PageRank and the score of the player with the lowest PageRank in each game.

Figure 2 is the corresponding picture, and it was confirmed that when the PageRank value is small, the score is generally lower than when the PageRank value is large. However, given that there are exceptions such as obtaining 8 points for a small PageRank or 0 points for a large PageRank, it was judged that it is not appropriate to simply consider only PageRank when predicting scores.

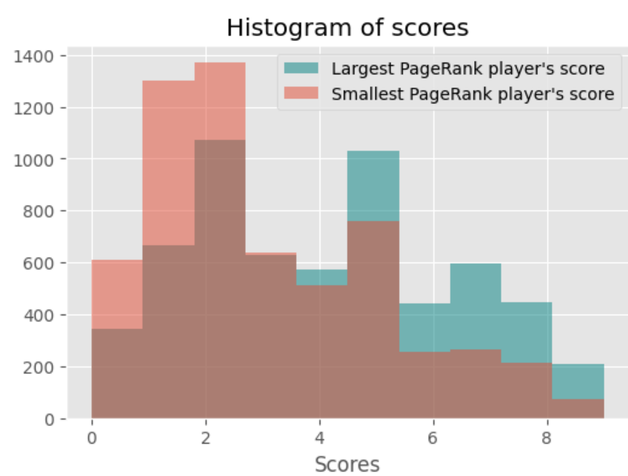

Figure 2: Distribution of scores based on PageRank

Finally, it was confirmed that the average score varied significantly depending on the number of players in the game. This will later be used for prediction along with average score and PageRank. Figure 3 shows the average **Y** of the game score according to the number of participants **X**.

Therefore, in addition to PageRank, we will use various characteristics such as average score and number of players participating in the game to better reflect the information in the graph, and we

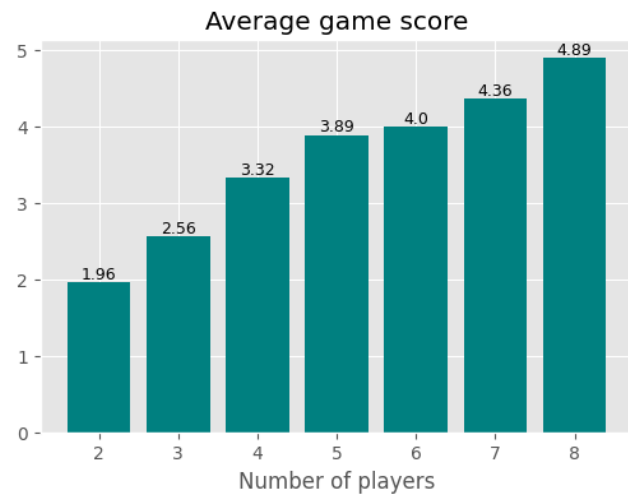

Figure 3: Average game scores based on the number of players

can expect an algorithm that performs better than simply predicting with the randomness or average user scores.

4 PROPOSED METHOD

4.1 Base algorithms

We used two base algorithms for two tasks.

i) GraphSAGE. GraphSAGE, which stands for Graph Sample and Aggregated Embeddings, is a graph representation learning algorithm that belongs to the family of Graph Neural Networks (GNNs) [1].

GraphSAGE is designed to learn embeddings (i.e., vector representations) for nodes in a graph, which capture the structural information and neighborhood relationships of those nodes. It is an inductive learning algorithm, meaning it can generalize to nodes not seen during training, making it particularly useful for large graphs with evolving or dynamic node sets.

For a target node, it samples a fixed-size neighborhood and aggregates features from neighboring nodes. The aggregated representation is then passed through a neural network to generate an embedding for the target node, capturing its structural and contextual information. GraphSAGE is widely used in applications involving graph-structured data and is valuable for tasks such as node classification, link prediction, and recommendation in large and complex graphs.

ii) PageRank. It is an algorithmic used by Google search engines to rank web pages in their search results. PageRank assigns a **rank** to each web page based on the number and quality of links pointing to it [2]. In other words, a web page is considered more important if it is linked to by other important pages.

PageRank is calculated iteratively. Each web page starts with an initial PageRank value, and in each iteration, the PageRank value of a page is updated based on the PageRank of pages linking to it. The more inbound links a page has, and the higher the PageRank of those linking pages, the more PageRank it receives.

PageRank is calculated using the following formula.

$$PR(A) = (1-d) + d\left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \dots\right) \tag{1}$$

PR - PageRank, d - damping factor, L - number of outbound links

4.2 Method 1: *Winner Prediction*

The main idea behind our method is to deal with different types of data with different features using heterogeneous graph(i) and GraphSAGE(ii).

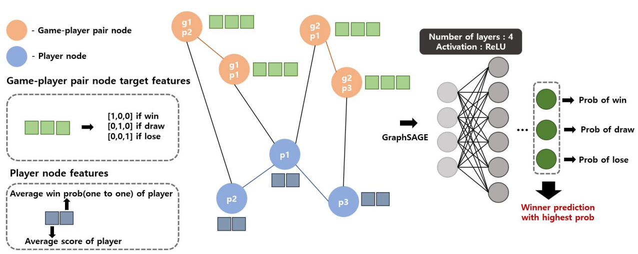

Figure 4: Winner prediction model overview

While traditional methods using one-node-type-graph are difficult to represent different types of features, **heterogeneous graphs** can represent diverse features and relationship efficiently. Therefore, our model can capture richer contextual information and produces better prediction outcomes. Also, by constructing 'game-player pair' node, we can predict the winner of each game efficiently by predicting the probability of resulting in win/draw/loose for each game-player pair node.

For the GNN, **GraphSAGE** is used for the inductive learning. It will generalize to nodes not seen during training, therefore can predict well for the validation and test sets as well.

For 'player' node features, two features were used, which are average score(1), and one to one game win rate(2) of each player.

i) Construction of heterogeneous graph.

- Node & Edge types - We first construct the undirected heterogeneous graph using all data in training, validation, and test sets. There are two types of nodes which are 'game-player pair' node and 'player' node. There are three types of edges which are between (game-player pair, game-player pair), (player, game-player pair), and (player, player).
- Game-player pair nodes - There are two 'game-player pair' nodes for each game, for example, ('game 1', 'player 1') and ('game 1', 'player 2') for game 1. Each game-player pair node has a three-dimensional target feature where each position means probability of resulting in win/draw/lose respectively.
- Player nodes - There are 'player' nodes for each player. They have two node features which are 'average score of the player in survival games' and 'average number of wins of the player in one-versus-one matches'.
- Connection rules - 'game-player pair' nodes are connected to each other if they share the same game. For example, ('game 1', 'player 1') node and ('game 1', 'player 2') node are connected. Each 'game-player pair' node is connected to the 'player' node who participated in the game. For example, ('game 1', 'player 1') node and 'player 1' node are connected. 'Player' nodes are connected to each other if they participate in the same game at least once. For example, 'player 1' node and 'player 2' node are connected.

ii) Training heterogeneous GNN with GraphSAGE.

- GNN consists of three GraphSAGE layers and one linear layer, a total of four layers. GNN gets the heterogeneous graph as an input and produces three-dimensional output.
- The loss is set as cross entropy between the output of the model and the actual label in three dimensions, which is a one-hot vector representing the winner of the game. Through training, the GNN model reduces loss and produces a three-dimensional output where each position represents the probability of resulting in win/draw/loose respectively. Finally, we choose the result with the highest probability for each game.

4.3 Method 2: *Score Prediction*

The main idea of score prediction is the generation of heterogeneous graph data(i) and learning using GraphSAGE(ii).

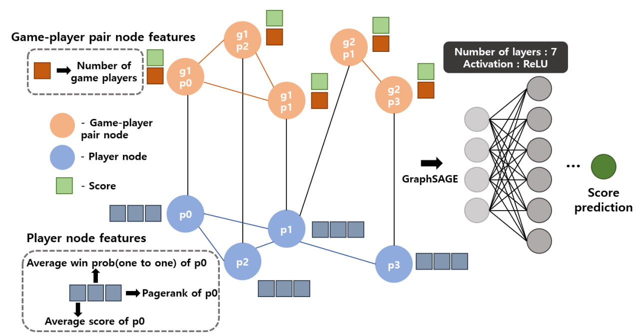

Figure 5: Score prediction model overview

As with winner prediction, a heterogeneous graph model was used to handle multiple types of nodes and features. Also, by constructing 'game-player pair' node, we can predict the score of each game efficiently by predicting the score of each game-player pair node.

In the generation of heterogeneous graph data, useful information including PageRank is calculated to construct node features, and in GraphSAGE, node features are learned using this information. Figure 5 is the model overview. The four elements that make up the node features of the heterogeneous graph are the number of players participating in the game(1), average score(2), one to one game win rate(3), and PageRank(4). All of these features were calculated using only the training data, and if the corresponding values could not be obtained, they were replaced with the average value.

To calculate PageRank, a weighted directed graph was created using training data. The graph created here is used solely to calculate PageRank and is different from heterogeneous graphs. The reason why PageRank was introduced in the score prediction problem was because the graph was created using the score difference as a weight, and it was assumed that the higher the PageRank value in the created graph, the higher the probability of obtaining a high score in the game.

The following is a specific method of designing a directed weighted graph to calculate PageRank.

- When there is a game in which N people participate, an edge is created in the direction from a person with a low score to a person with a high score, equal to the number of $\binom{n}{2}$.
- An edge weight is given equal to the score difference between high score and low score. If the scores are the same, the weight is set to 1.

i) Construction of heterogeneous graph.

- Node & Edge types - We first construct the undirected heterogeneous graph using all data in training, validation, and test sets. There are two types of nodes which are 'game-player pair' node and 'player' node. There are three types of edges which are between (game-player pair, game-player pair), (player, game-player pair), and (player, player).
- Game-player pair nodes - There are as many 'game-player pair' nodes as number of person participating for each game. For example, when 3 people participate in game 1, there are ('game 1', 'player 0') and ('game 1', 'player 1') and ('game 1', 'player 2') game-player pair nodes. Each game-player pair node feature has a one-dimensional feature that represents the number of players participating in the game, and the one-dimensional target feature represents the player's score in the game.
- Player nodes - There are 'player' nodes for each player. They have three dimensional node features which are 'average score of the player in survival games' and 'average number of wins of the player in one-versus-one matches' and 'PageRank in directed weighted graph'.
- Connection rules - 'game-player pair' nodes are connected to each other if they share the same game. For example, ('game 1', 'player 0') node and ('game 1', 'player 1') node are connected. Each 'game-player pair' node is connected to the 'player' node who participated in the game. For example, ('game 1', 'player 0') node and 'player 0' node are connected. 'Player' nodes are connected to each other if they participate in the same game at least once. For example, 'player 0' node and 'player 1' node are connected.

ii) Training heterogeneous GNN with GraphSAGE.

- GNN consists of 7 GraphSAGE layers and one linear layer, a total of 8 layers. GNN gets the heterogeneous graph as an input and produces one-dimensional output.
- The loss is set as mean squared error between the output of the model and the actual label in one dimension, which represents score of each player of the game. Through training, the GNN model reduces loss and produces an output close to the actual label.

5 EXPERIMENTS

5.1 Experiment 1: *Winner Prediction*

To predict the outcome of each game, we generated a heterogeneous graph with all training, validation, and test sets. Then, we trained the GNN model only with training set. Figure 6 shows the graph of training loss and training/validation accuracies for 500 epoch. Loss decreases as training proceeds and both training and validation accuracy increase well. Our model shows a validation accuracy of nearly 80 which is much higher than other methods (Table.1).

Method	Validation accuracy
Random guessing	50.48
Counting	67.33
Heterogeneous graph with GraphSAGE	**79.70**

Table 1: Result of Winner Prediction

Figure 6: Loss and accuracy of winner prediction

5.2 Experiment 2: *Score Prediction*

To predict the scores of each game, we first used PageRank and average score methods. This method calculates the score by using the average score considering the number of players in the game and giving additional points to the person with the highest PageRank among the players in each game. When combining PageRank to average score, it shows approximately 4.5 percent higher validation accuracy(84.55) than when using only average score method(80.08). Next, we generated a heterogeneous graph with all training, validation, and test sets as in score prediction. Then, we trained the GNN model only with training set. Figure 7 shows the graph of training loss and training/validation accuracies for 300 epoch. Loss decreases as training proceeds and both training and validation accuracy increase well. Our model shows a validation accuracy of about 87 which is much higher than other methods (Table.2).

Method	Validation accuracy
Random guessing	70.49
Average score	80.08
PageRank & average score	84.55
Heterogeneous graph with GraphSAGE	**86.79**

Table 2: Result of score prediction

Also, in order to find the optimal structure in the constructed heterogeneous graph network, the validation accuracy was compared for each number of GraphSAGE, and the best validation accuracy was found in 7 layers. (Table.3) Considering that the highest performance was achieved in 7 layers, there are as many game-player pair nodes and player nodes as the number of participants in each game, so it can be expected that the greater the number of layers, the easier it is to transfer information.

Method	Validation accuracy
Layer 3	86.46
Layer 5	86.43
Layer 7	**86.79**
Layer 9	86.75

Table 3: Validation accuracy according to number of layers

Figure 7: Loss and accuracy of score prediction

6 CONCLUSIONS

We predicted result in battle games by three main graph methods. For the winner prediction in one-versus-one game and score prediction in battle royal matches, we innovatively employed a heterogeneous graph with two types of nodes to represent the battle game in graph. Then, we trained the GNN model with GraphSAGE layers to inductively learn the structure of battle game graph. To add more information in graph, we used PageRank and average scores for the node features. In addition, the number of players in the game was also meaningful information in predicting scores.

The introduction of the 'graph-player pair' node, as opposed to the conventional 'game' and 'player' nodes, proved pivotal in representing relationships and actual labels in a more original and efficient manner. This innovative node type facilitated the capture of richer contextual information, enabling our model to discern patterns and relationships that were previously challenging to uncover using traditional methods.

A notable feature of our approach was the incorporation of additional information into the graph. By leveraging PageRank and average scores as node features, we enriched the contextual landscape, providing our model with a more comprehensive understanding of the intricate interplay between game elements. Furthermore, recognizing the significance of the number of players in each game, we incorporated this crucial information into our predictive model, contributing to its enhanced accuracy in score predictions.

By adding rich information to the graph neural network, our model learned the deep context and pattern in the battle game graph. The superior performance achieved by our approach, compared to traditional methods, highlights the efficacy of leveraging graph structures and enriched node features for game outcome prediction.

It's important to note that our research was conducted with a limited dataset containing only game and outcome data. Looking ahead, we envision that incorporating more diverse data, such as individual player attack patterns and environmental information specific to battle games, will further elevate the predictive performance of our model. This future research direction holds the promise of unlocking deeper insights and pushing the boundaries of predictive accuracy in the dynamic realm of battle games.

REFERENCES

[1] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

[2] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998. Paper SIDL-WP-1999-0120 (version of 11/11/1999).

A APPENDIX

A.1 Labor Division

The team performed the following tasks

- Devise the overall method [ALL]
- Winner prediction [Dayeon Jeong]
- Score prediction [Yunbyeong Chae]

A.2 Full disclosure wrt dissertations/projects

Dayeon Jeong: She is not doing any project or dissertation related to this project: her thesis is on subtyping of Parkinson's Disease.

Yunbyeong Chae: He is not doing any project or dissertation related to this project: his thesis is on inferring cell-cell communication using spatial transcriptomics.