League of Legends Network Analysis

Group 4: Shikuan Gao, Ruiwen Shi

# Introduction

League of Legends is a multiplayer online battle video game developed and published by Riot Games in 2009. In League of Legends, players assume the role of an unseen "summoner" that controls a "champion" with unique abilities and battle against a team of other players or computer-controlled champions. The goal is usually to destroy the opposing team's "nexus", a structure which lies at the heart of a base protected by defensive structures. In the last ten years, League of Legends officially becomes the most popular E-Sport. Not only it is the most played PC games in the world, but also set up many professional leagues in many regions such as in North America, Europe, Korea and China. Riot games also set up world champion series once a year, it is the most important competition in E-Sports. According to the statistic, last year it has over 33 million audience watches the whole championship.

## The rising of E-sports

Since in 2009, League of Legends has evolved from a small population of desktop-computer warriors into a full-scale phenomenon. League of Legends only started with fans organizing tournaments. In the process, [it has become an e-sport](http://www.nytimes.com/2014/10/12/technology/riot-games-league-of-legends-main-attraction-esports.html). Riot grafted the League of Legends Season one championship in 2011. After about ten years, it brought in $1.5 billion total revenue in 2017, could hit $2.3 billion by 2022,

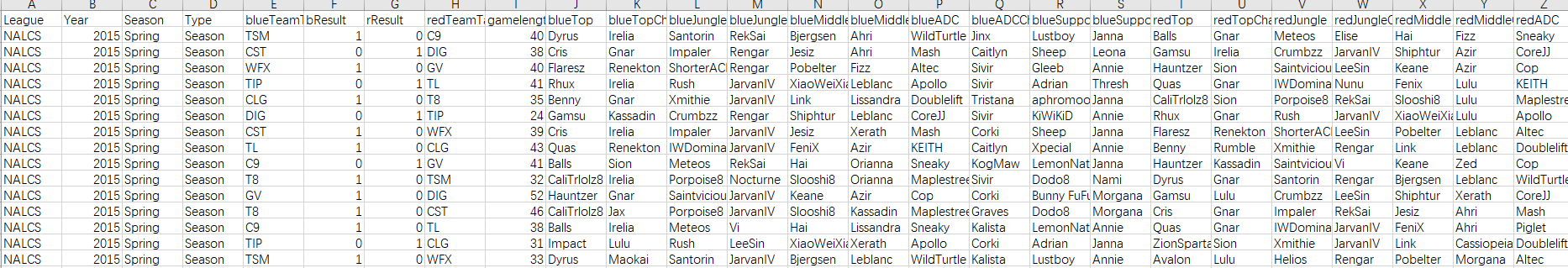
# Objective

As the rising of E-sports in these years, more and more revenue are made by E-sport matches and relevant activities. League of Legends, as the most popular E-sport in the world, can affect our lives more than we believe. In this project report, our goal is to generate a visual network for league of legends for season 2017 from several aspects such as powerful teams, game style of teams, special team and so on. The outcome of our results should be valuable for the pro league of legends teams and make suitable analysis for each unique pro team for proper adjustment on team plans.

# Data

### *Data Description*

The raw data of our project is from Kaggle, it is a dataset for season 2015-2017 match information. By checking the dataset in the below of figure 1, we can see that there are over 7000 match variables and all the attributes we needed such as result, champion selected, game length. Each attribute influences on the outcome of a single match.



**Figure 1. Dataset**

### *Data Cleaning*

### Get all matches information from 2017

First, in order to generate a visual network for league of legends for season 2017. We need to get all the matches information from 2017. From the figure 2 shown below, we use python code set index to be ‘Year’. The next step is to use the python code ‘. loc’ to find all the index ‘Year’ to be ‘2017’.A screenshot of a cell phone

Description generated with very high confidence

**Figure 2.**

### Get the winning side champions selected

Second, from all the 2017 matches, we want to select all the winning side five champions. From the figure 3 and figure 4 shown below, we use python code set index to be ‘rResult’ and ‘bResult’. And we use python code ‘.loc’ to find all the red team champions selected when ‘rResult’ is 1 and to find all the blue team champions selected when ‘bResult’ is 1. Note from our data set, Result from each side equal side means this side is winning.A screenshot of a cell phone

Description generated with very high confidence

**Figure 3. Blue team wining champions**

A screenshot of a cell phone

Description generated with very high confidence

**Figure 4. Red team wining champions**

### Cross table

After we have all our winning side champion pool. Next step is to use the dataset to generate an adjacency matrix for the future use of our network model. In this step, from the figure 5 and figure 6 shown below, we use python code ‘pd.crosstab’ to get the cross table of middle/jungle and ADC/support champions.

A screenshot of a cell phone

Description generated with high confidence

**Figure 5. Cross table of middle/jungle**A screenshot of a computer

Description generated with high confidence

**Figure 6. Cross table of ADC/support**

### Adjacency matrix

After generating the cross table, the next step is to use the cross table to generate the adjacency matrix for our network. The adjacency matrix is a matrix of ones and zeros where a one indicates the presence of a connection. And the numbers in adjacency matrix can be shown as the direction and relation between two nodes in the network. From the figure 7 and figure 8 shown below, we use python code ‘pd.crosstab’ to get the adjacency matrix, and use python code ‘columns.union’ to get all the index of the cross table, finally, we use python code ‘reindex’ to set the index of adjacency matrix with fill value equal to 0.A screenshot of a computer

Description generated with very high confidence

**Figure 7. Adjacency matrix of middle/jungleA screenshot of a computer

Description generated with high confidence**

**Figure 8. Adjacency matrix of ADC/support**

### Game length

To generate our network model, one of the most important weight we are going to use is the game length, in a match when game length is shorter it means that the competition is unequal. In this step, we are going to get game length information for each match from winning team directed to losing team. From the figure 9 shown below, we use python code set index to be ‘bResult’. And we use python code ‘.loc’ to find all the team versus and game length.

A screenshot of a cell phone

Description generated with very high confidence

**Figure 9. Game Length Data**

# Exploratory Data Analysis

In [statistics](https://en.wikipedia.org/wiki/Statistics), exploratory data analysis (EDA) is an approach to [analyzing](https://en.wikipedia.org/wiki/Data_analysis) [data sets](https://en.wikipedia.org/wiki/Data_set) to summarize their main characteristics, often with visual methods. A [statistical model](https://en.wikipedia.org/wiki/Statistical_model) can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task.

### Histogram

In our project, we want to focus one of the most important features in our dataset, game length, and to do the EDA of it. First, we make a histogram (Shown from figure 10). The traditional histogram count is on the left y axis and the percentage of all games is on the right y axis. It shows the most frequent minute value of all games played. We can see that the graph is slightly skewed right (tail of the data is longer on the right). Also, the outlier game lengths around the 60-minute mark.

A close up of a map

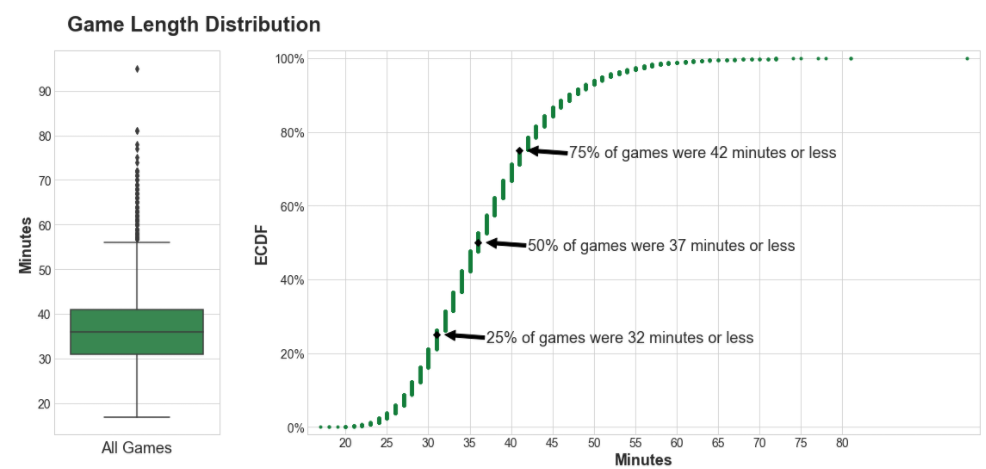
Description generated with very high confidence

**Figure 10. Histogram**

### Box Plot and ECDF (Empirical Cumulative Distribution Function)

Second, we make a box plot. From the graph it shows that the box plot is the most intuitive way to show our data. The green shaded part is known as the IQR (inter quartile range) and each horizontal line of the box represents the 25th, 50th, and 75th percentiles from bottom to the top. Also, in the box plot there is a context here is the outliers, it shows as the little dots outside of the shaded part and give us the view of how our data looks like.

Last, we make an ECDF (Empirical Cumulative Distribution Function) graph. It is an insightful way of seeing what percent of the variable data is at or below a particular value. As the mentioned in the first plot show, 25% of the game length data is at 32 minutes or less. Also, move up and to the right of the ECDF and it will continue to mention the Y% of game length data that is at or below X minutes.



**Figure 11. Box Plot and ECDF**

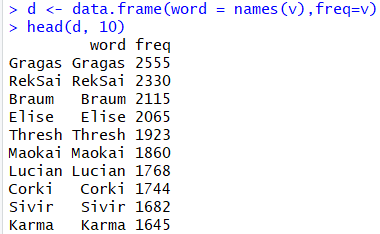
# Data Visualization – Word Cloud

In the data visualization, we first use R to make a word cloud to see visual representation of our data. In the word cloud, tags are usually single words, and the importance of each tag is shown with font size or color. In our word cloud, the size represents the number of times that tag has been applied to a single item. This is useful as a means of displaying [our](https://en.wikipedia.org/wiki/Metadata) data about an champion that has been [shown](https://en.wikipedia.org/wiki/Democracy) in a winning team for the last three years.



**Figure 12. Word Cloud**

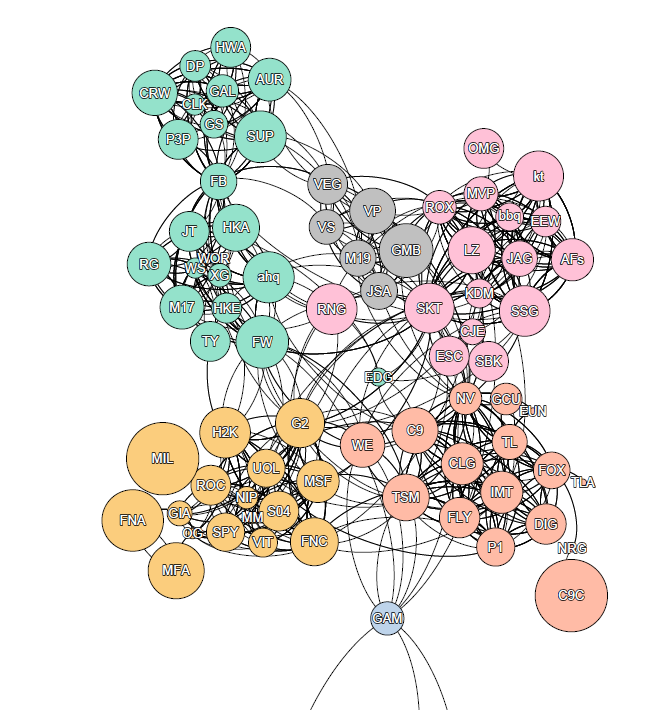
From the figure 12 shown as above, it shows that the most selected champion in the winning side for last three years. Also, there is a list we use R to show the top 10 selected champions shown as below as figure 13. From the list we see that these 10 champions may not be too strong to get them be banned in the beginning stage, but not too weak to not show up in the game. They have their own unique function in the game that makes them the top 10 selected champions.



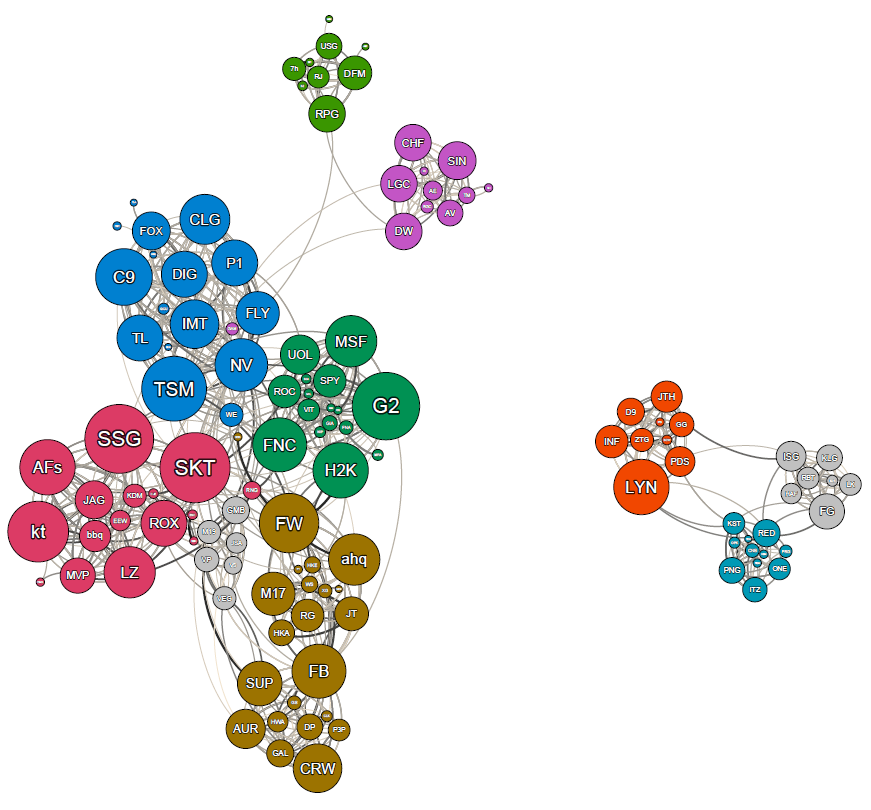
**Figure 13**

# Data Visualization – Network

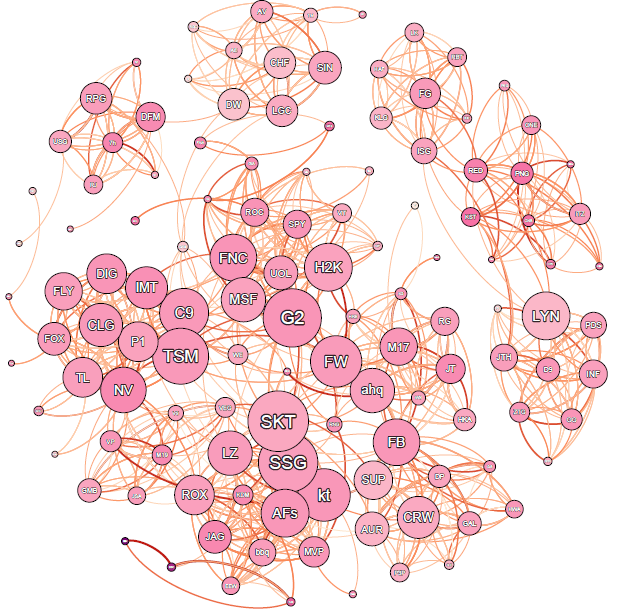
With the game length data that we acquired above, we import it as an edge table into Gephi to establish our social network. So, the basic indicator of the network includes: nodes – each team, edges – matches (directed: source - win team, target - lost team), weight - game length, degree - the counts of matches that one team had played, weighted degree - total game length of each team, win rate = out-degree/degree (attribute created), average game length = weighted degree/degree (attribute created). And, when we import the table into the software, we chose to allow multiple edges between two nodes, as teams could play against each other repeatedly.



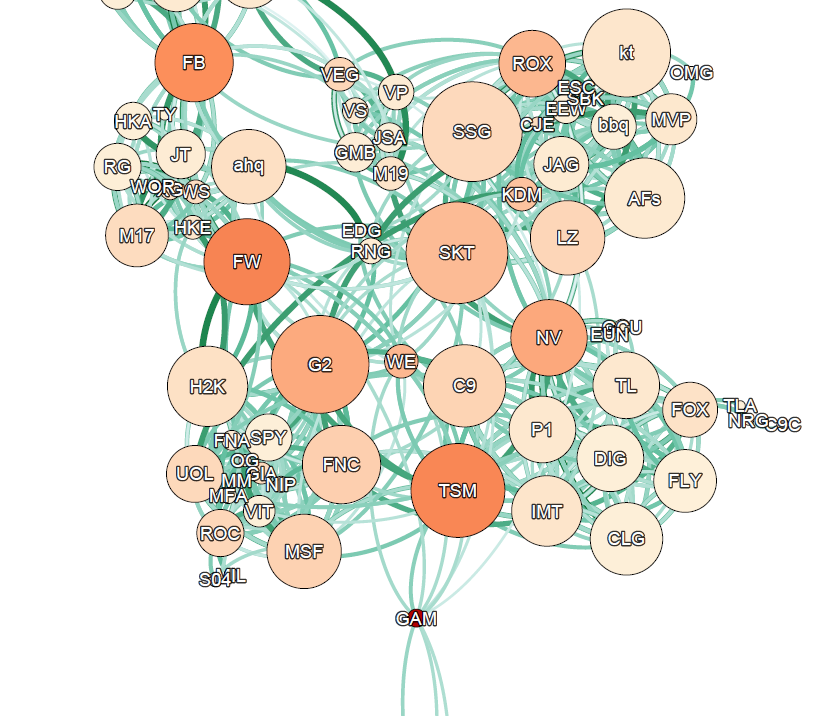
Firstly, we set modularity as the color of the nodes and win rate as the size of the nodes. As the graph shows, teams were separated by their colors into modules nearly in accordance with the divisions of league around the globe. Exceptionally, for the lack in the data of LPL (the Chinses league), Chinese teams like WE, EDG, RNG which appeared in the World Champion were assigned to different modules. From the size of the nodes, the obvious bigger nodes like MIL, FNA and C9C were the noises because the matches they played were rather less than the normal level, mostly below 5 matches. And the win rate between strong teams, like SKT, SSG, and normal teams are not so differential in this graph. Hence, win rate is not a prominent index to indicate the performance of a team during the season.



Then, instead of win rate, we set the out-degree (the win counts of each team) as the nodes’ size. This graph is much better than the previous one in displaying the strong teams like SKT, SSG, TSM. Because the teams that performance better would advance in a tournament and win most of the games.



Next, we try to dig in into the playing style of each team by using the game length attribute. We set node color as average game length and node size as out-degree to observe the relation in the game length of strong teams. In addition, we set edge color as weight to see the distribution of game length in the season. The graph shows no significant difference in average game length of each team, although there indeed is deviation in each game. No matter the teams played aggressively or not, most matches would end around 30-40 minutes. This situation is probably due to that in the game, each size would have constructions called turrets. The turrets are extremely powerful in the early game, however, around 30 minutes, with items to enhance the champions, the turrets become much easier to destroy. The designers of this game set up these mechanisms to control the game length between this interval, which they believe would be the best watching experience for the audiences.



Finally, we set node color as betweenness centrality to find out the relation between performance of the team and their importance in the league. The graph successfully shows powerful teams in each division, as they performed better and had more chances to play against other division, like the World Champion. However, there is one exceptional, team GAM. The GIGABYTE Marines (GAM) surprised everyone at the 2017 Mid-Season Invitational when they not only survived the Play-In, but thrived in the Group Stage, taking wins off some of the biggest teams in the world. They earned the GPL a Group Stage seed at Worlds and were the rightful ones to take it as they reclaimed the title of GPL Champions with their victory over Thailand’s Ascension Gaming. Their hyper aggressive playstyle will make them one of the most exciting teams to watch at Worlds 2017. This team becomes so important in this network because it mainly played in the secondary league and survived the Play-in stage of World Champion to obtained the chance to play against higher level league opponents.

# Conclusion

Win counts performs better than win rate when illustrating the power of a team.

No significant difference founded in average game length among most teams. Most matches ended between 30 - 40 mins.

Betweenness centrality shows powerful teams within each module, as they would be directly invited to represent their own division.

However, GAM is an exception, marked by betweenness, as the intermediary between secondary league and major league.

# References

2017 World Championship

<https://esc.watch/tournaments/lol/2017-world-championship>

Kaggle Data Source

<https://www.kaggle.com/chuckephron/leagueoflegends/data>

ggnet2: network visualization with ggplot2

<https://briatte.github.io/ggnet/>