**League of Legends Match Outcome Prediction using Deep Learning Techniques**

James Ting

May 22, 2020

**Introduction**

League of Legends is a popular and competitive Multiplayer Online Battle Arena, where 2 teams of 5 players compete to destroy the enemy Nexus (Figure 1a & Figure 1b). Each player controls a champion (a character), and to win, the team must destroy a series of enemy turrets, invade the enemy base, and destroy the opponent’s Nexus.

As the players play each match, their champion will level up and earn gold, which the player can then exchange at the shop for items that can strengthen each character. Players also earn gold from: kills, killing sprees, enemy minions, and neutral monsters.

Throughout the map, there are buffs that each team can earn by killing neutral monsters. These neutral monsters also provide gold to the player who earns the kill, which allows the players to obtain more items.

League of Legends has, at the time of writing, approximately 120 million monthly players. Furthermore, Riot Games (the company behind the game) has an API that anyone can use in order to pull data about any player, and their match history. As such, there is a massive wealth of data that is available and makes it perfect for a machine learning project. **In this project, I will use a binary classification neural network to predict match outcomes on two types of data: pre-match data and post-match data**. Pre-match data will be data that is available prior to the start of the match, and post-match data will also include each player’s performance data for this specific match.

**Data Collection**

The data was collected using the RiotWatcher Python Library (RiotWatcher, 2019). I built classes to help facilitate the collection of data. The first class, the AbstractDataPuller (and it’s sub classes) will be the objects that will make calls to the RiotWatcher library to get the information from the API. It contains three subclasses: PlayerDataPuller, MatchDataPuller, ChampionMasteryDataPuller. Each subclass is responsible for pulling data for players, matches, and champion mastery respectively.

The next class is the MatchCrawler class. It is essentially an iterator, that has 2 functions: hasNext() and next() (similar to a Java Iterator Object). hasNext() simply returns a boolean representing whether the specified number of data points has been added to the data set. next() returns the next matchID to pull the data for. next() selects the next match ID by taking the current match ID, and for a random summoner in that match, gets a non-empty match list. If that match list is empty, then it moves to the next summoner. Once a non-empty match list has been found, it will return a random matchID in the list.

The final class is the DataSetMaker class. This class makes use of the two classes, and writes each data points to the file appropriately.

**Data Preparation**

**The Network**

**Results**

**Conclusion**



Figure 1a. Summoner’s Rift Map. The 2 Nexus’ can be seen in the top-right and bottom left corners of the map



Figure 1b.: The blue team Nexus(Nexus, 2020)

**References:**

pseudonym117. 2019. *Welcome to RiotWatcher’s Documentation! – RiotWatcher 3.0.0 documentation*. [online] Available at: < <https://riot-watcher.readthedocs.io/en/latest/>> [Accessed 22 May 2020]

League of Legends Wiki. 2020. *Nexus*. [online] Available at: <<https://leagueoflegends.fandom.com/wiki/Nexus> > [Accessed 22 May 2020].

League of Legends Wiki. 2020. *Summoner’s Rift*. [online] Available at: <<<https://leagueoflegends.fandom.com/wiki/Summoner's_Rift>> [Accessed 22 May 2020].