Data-driven Recommendation Systems for Multiplayer Online Battle Arenas

Rohit Bhattacharya
Johns Hopkins University - Computer Science
rbhatta8[at]jhu.edu

Azwad Sabik
Johns Hopkins University - Robotics
asabik2[at]jhu.edu

Abstract

In this paper, we present a data-driven player-personalized recommendation system for League of Legends. We are able to identify 14 distinct clusters of playstyles that a player can fall into in the current metagame. Based on these clusters we generate user-specific recommendations for champion selection and gameplay behaviours for distinct temporal segments of a match.

1 Introduction

Over the last two decades, the world has witnessed a dramatic rise in the popularity of eSports, and in particular, Multiplayer Online Battle Arenas (MOBAs). The two most representative games of this genre today are Dota 2 and League of Legends, accounting for over 30 million active players on a daily basis[13][15]. Prize pools have risen from pittances to $6.6 million[4] earned by the winners of the latest edition of Dota 2's flagship tournament, The International, making gaming a viable, even lucrative career in today's day and age.

The reality, however, is that these games are known to have steep learning curves, and once a certain skill level is achieved, many players find themselves unable or unsure of how to go about improving their gameplay. With the current popularity of these games, and the potentially limitless stream of data that can be obtained from them through convenient APIs, it should come as no surprise that MOBAs makes for a very interesting "toy" problem, albeit a very serious one, in the eyes of many. In this paper, we aim to provide data-driven, personalized recommendations of character selection and suggestions to improve individual gameplay, to individuals playing the game, League of Legends (LoL), created and managed by the developer, Riot.

More concretely, we look to identify clusters of different playstyles at the very highest level of play in LoL, that is, the Challenger League, consisting mostly of professional players. This enables us to then look back on a novel player's matches, identify his/her most "successful" cluster, and further present champion recommendations and "habits" of professional players that were observed in the same. Thus, we aim to improve an individual's gameplay, without necessarily forcing them to switch from a playstyle that they have already seen some measure of success with.

1.1 Gameplay Overview

A MOBA is a 5v5 real time strategy game, where players control a single character, or "champion," (from a pool of 128 in the case of LoL) for the entirety of a match that typically lasts about 40 minutes. In addition to player-controlled champions, the game also incorporates various simple AIs such as minions for each faction that spawn in waves, faction-specific "turrets" which provide stationary defenses, neutral creatures that reside in the jungle areas, and elite neutral units, that when killed, provide great rewards to the faction that killed it. The objective is to fight through hostile units and destroy the enemy base while defending your own; all the multitude of decisions made in-between make up the complex gameplay of a traditional MOBA.

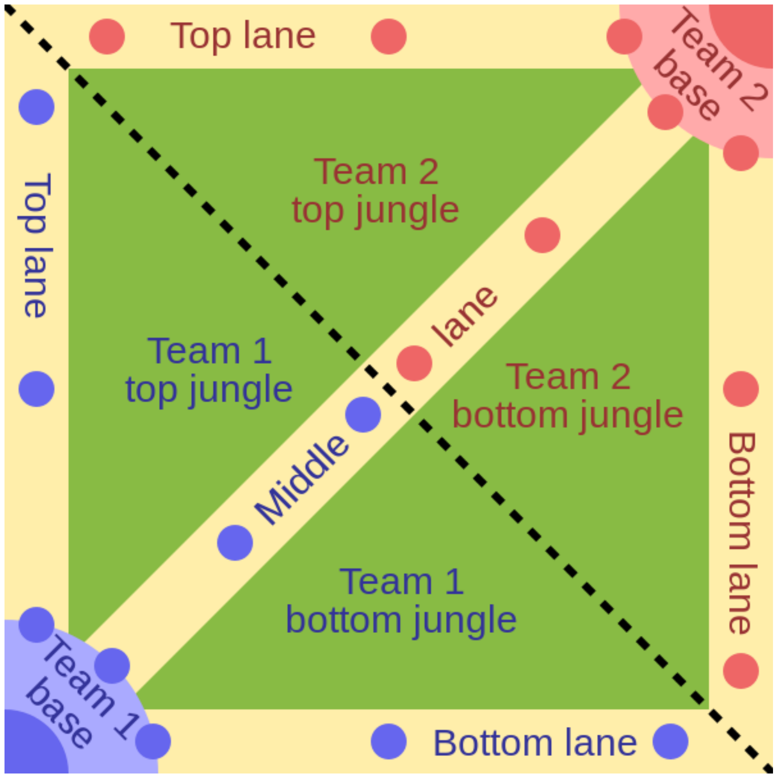


Figure 1: The general layout of a typical MOBA game

The champions available to a player can perform a variety of combat-related activities such as dealing “physical” or “magic” damage, healing themselves and other champions, affecting the stats and mobility of other units, or even taking damage on behalf of teammates. These activities are the result of low-level commands initiated via mouse-clicks and keypresses for the purposes of movement, ability use, and item use, and provide the basis of methods by which a player can interact with the game’s environment, including all other units and buildings. It is important to note that these activities and their targets can be prioritized by a player based upon the role suited to the player’s champion or necessary to the team’s overall functionality, because these priorizations are the in-match manifestation of the “playstyles” analyzed in this paper.

As a simple example, a player that chooses to play his champion in a supporting role might spend less time gaining gold and experience from killing enemy minions and neutral units and instead prioritize setting up kills on enemy champions or thwarting ambushes set up by the enemy team. In order to assist new players, who are learning the game, Riot has established “tags” for each champion that defines roles that they are suited to play. Very often, it is seen however, that professionals and long-time players can “break the meta”, so to speak, and define their own unique ways of playing the game.

In this paper, we are particularly interested in various game-defining behaviours such as, physical or magic dealt, damage taken, minions killed, enemy champions slain, and turrets destroyed, by a player’s champion at the end of the game as well as timeline data concerning these statistics.

2 Related Work

Several websites[5][9] for examining statistics of character trends and win rates exist for Dota 2 and LoL. There are also applications that provide more algorithmic approaches to the analysis of these games. These include ones such as, Dota Picker[14] and our very own award-winning, Winsight[3], that help counter overall strategies by suggesting "counter-picks" to the enemy team's lineup. There also exists a more recent machine learning oriented application that uses neural networks and genetic algorithms to find optimal "jungling" routes in Dota 2[1].

3 Methodology

In this section, we will describe some of the methods we utilized to analyze our data. Implementations for all of the algorithms mentioned below are readily available through the Python machine learning package scikit-learn[6] and were used out-of-the-box for our analysis. This project's pipeline implementation code can be found at its associated github repository[2].

3.1 Data Acquisition

End-game statistics and timeline data was extracted (using a Python wrapper[12] for the Riot API[10]) for 500 matches in the highest tier of competition in League of Legends, the Challenger league. Each of these matches contains 5 winners (the winning team) and 5 losers. We extract the relevant statistics only for the winners of each match (as this indicates to us a strategy that was successful), resulting in 250 "training" examples, and their features. In addition to this, we also extracted 5 matches worth of data for a few other players that were not in the previous set, in order to test our recommendation system. Further, we also extracted the recommended primary tags assigned by Riot for each champion, indicating preferable roles to put the champion in. This was done to validate the clustering we perform on the data set later in the pipeline.

3.2 Dimensionality Reduction

In order to help us visualize the data, and reduce the computation time for later clustering algorithms, we used various unsupervised techniques of dimensionality reduction. To better understand and identify distinct playstyles in the data, data points were coloured using champion role tags as defined by Riot as well as labels produced by our own clustering. Note that the tags provided by Riot only served as a visual aid when validating our results, not as the ground truth for any clustering that we performed.

3.2.1 Principal Component Analysis (PCA)

PCA[8] is a highly popular technique used for dimensionality reduction. It performs a linear mapping of the data to a lower-dimensional space such that the variance of the data in the low-dimensional representation is maximized. Results were visualized for projections onto the top 2 as well as top 3 components of PCA.

3.2.2 Kernel PCA

The technique of PCA described in the previous section can be non-linearized by using the "kernel" trick. We experimented with three different kinds of kernels - 2^{nd} degree polynomial, 3^{rd} degree polynomial, and the radial basis (rbf) kernel.

3.2.3 Locally Linear Embedding (LLE)

LLE is a non-linear method of dimensionality reduction that preserves the neighbourhood embeddings of high-dimensional data[11]. The algorithm can be tuned on the number of neigbours used to reproduce each point in the lower-dimensional space. For our purposes, we used the recommended setting of 30 neighbours, and upon preliminary examination this appeared to provide us with favourable results. In addition, similar to PCA, we visualized projections of the data onto the top 2 as well as top 3 components.

3.3 Clustering

Clustering via affinity propagation[7] was performed on the raw end-game data as well as the PCA and LLE projected data. Affinity propagation is a clustering algorithm that is based on the notion of "passing messages" between data points in order to find a good set of "exemplars" that represent each cluster. It is important to note that we chose this algorithm in particular because it does not require a pre-determined estimate of the number of clusters as an input, and is unsupervised i.e. no pre-defined tags or labels (which, in our case, might have been the primary tags assigned by Riot) are used in order to minimize an objective. In terms of tuning, the recommended gamma of -50, and 1000 max iterations seemed to produce promising preliminary results.

3.4 Recommendation

A system was created for generating both champion and playstyle recommendations for a novel player given the player's match history.

3.4.1 Champion Recommendation

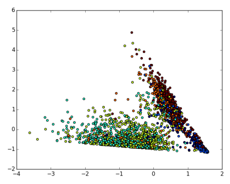
The underlying assumption of champion recommendation is that functionally similar champions will be found together in the low-dimensional clusters. The developed process for recommending champions converts a novel player's functional performance metrics from all of the player's previous match wins into the space of reduced dimensionality via projection and then minimizes pairwise distances between each of those projections and the previously identified professional clusters in order to identify which cluster most closely approximates player's most successful match-level playstyle, where a player's success with a particular cluster is approximated by the number of wins achieved using the playstyle associated with that cluster. Once a player's "most successful cluster" is found, champion recommendation is based upon the properties of the cluster itself. Given that a champion was associated with each of the professional match-wins that consists the points in each cluster, a simple ordered list of champions prioritized by their professional success in the cluster can be provided to the user to ultimately recommend champions both compatible and effective given the player's most successful playstyle successful.

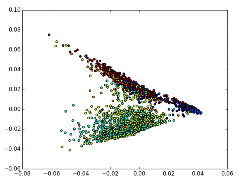
3.4.2 Playstyle Recommendations

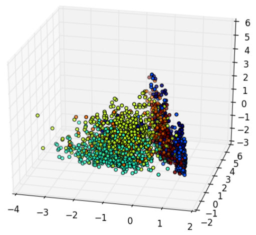
The Riot API's provision of match event timelines and each players' functional performance metrics for ten-minute segments of all matches makes it possible to generate recommendations for how a player should behave for each ten minute segment of a novel match in order to increase his/her likelihood of winning. Our method utilized regression based on partial least squares, which finds transformations of data from two different views based upon maximization of the transformed data's covariance. Recommendations for a player's playstyle for the first ten-minute segment of a given match utilized their most successful cluster's cluster center to provide target end-game data as the input to a regression between professional endgames in the cluster and professional first segments - this allowed recommendation of performance goals for an ideal first segment aiming to achieve the given playstyle. Recommendations for a player's subsequent segments (n) in a novel match can be achieved similarly, utlizing professional in-cluster data to generate predictions for segment-n's responses given segment-n_{-1}'s inputs.

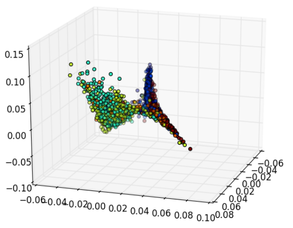
4 Results

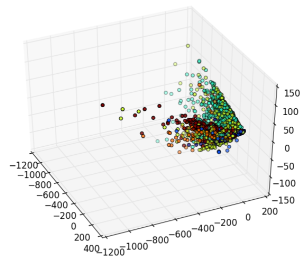
4.1 Dimensionality Reduction


(a) PCA 2 components


(b) LLE 2 components


(c) PCA 3 components


(d) LLE 3 components


(e) KPCA - Polynomial degree 2 Kernel

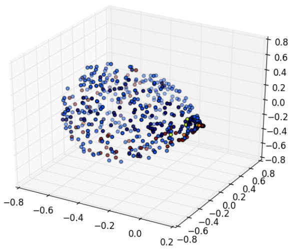

(f) KPCA - RBF Kernel

Figure 2: Plots of dimensionality reduction, colours are assigned based on Riot recommended champion tags

4.2 Clustering

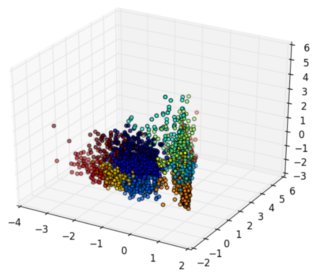
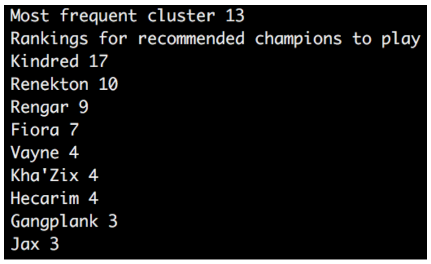


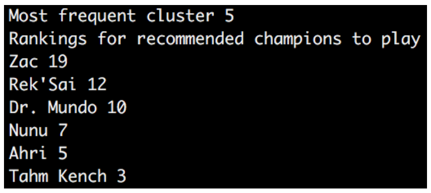
Figure 3: Affinity propagation clustering in PCA space

4.3 Recommendation



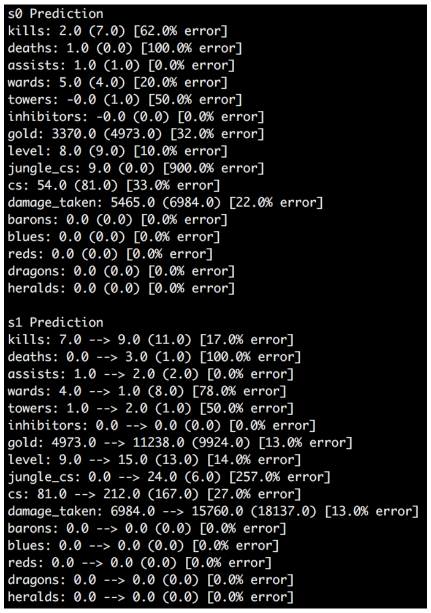
```
Most frequent cluster 13
Rankings for recommended champions to play
Kindred 17
Renekton 10
Rengar 9
Fiora 7
Vayne 4
Kha'Zix 4
Hecarim 4
Gangplank 3
Jax 3
```

(a) Champion recommendations for C9 Sneaky



```
Most frequent cluster 5
Rankings for recommended champions to play
Zac 19
Rek'Sai 12
Dr. Mundo 10
Nunu 7
Ahri 5
Tahm Kench 3
```

(b) Champion recommendations for JayJ



```
s0 Prediction
kills: 2.0 (7.0) [62.0% error]
deaths: 1.0 (0.0) [100.0% error]
assists: 1.0 (1.0) [0.0% error]
wards: 5.0 (4.0) [20.0% error]
towers: -0.0 (1.0) [50.0% error]
inhibitors: -0.0 (0.0) [0.0% error]
gold: 3370.0 (4973.0) [32.0% error]
level: 8.0 (9.0) [10.0% error]
jungle_cs: 9.0 (0.0) [900.0% error]
cs: 54.0 (81.0) [33.0% error]
damage_taken: 5465.0 (6984.0) [22.0% error]
barons: 0.0 (0.0) [0.0% error]
blues: 0.0 (0.0) [0.0% error]
reds: 0.0 (0.0) [0.0% error]
dragons: 0.0 (0.0) [0.0% error]
heralds: 0.0 (0.0) [0.0% error]

s1 Prediction
kills: 7.0 --> 9.0 (11.0) [17.0% error]
deaths: 0.0 --> 3.0 (1.0) [100.0% error]
assists: 1.0 --> 2.0 (2.0) [0.0% error]
wards: 4.0 --> 1.0 (8.0) [78.0% error]
towers: 1.0 --> 2.0 (1.0) [50.0% error]
inhibitors: 0.0 --> 0.0 (0.0) [0.0% error]
gold: 4973.0 --> 11238.0 (9924.0) [13.0% error]
level: 9.0 --> 15.0 (13.0) [14.0% error]
jungle_cs: 0.0 --> 24.0 (6.0) [257.0% error]
cs: 81.0 --> 212.0 (167.0) [27.0% error]
damage_taken: 6984.0 --> 15760.0 (18137.0) [13.0% error]
barons: 0.0 --> 0.0 (0.0) [0.0% error]
blues: 0.0 --> 0.0 (0.0) [0.0% error]
reds: 0.0 --> 0.0 (0.0) [0.0% error]
dragons: 0.0 --> 0.0 (0.0) [0.0% error]
heralds: 0.0 --> 0.0 (0.0) [0.0% error]
```

(c) PLS regression recommendations for C9 Sneaky

Figure 4: Examples of recommendations output by our system

5 Discussion

5.1 Dimensionality Reduction

PCA and LLE produced similar representations of the end-game statistics when used to project the data onto the top two most salient components identified by each, as can be seen in 2a and 2b respectively. Utilizing a non-linear kernel did not appear to provide any advantage in identifying useful three-dimensional bases, with a 2^{nd}-order polynomial projection producing the most structured representation of the data, the 3^{rd}-order polynomial producing less, and RBF projections producing nebulous clouds of data. The structure of the data differed significantly when projected onto three components by each method, with all points appearing more "tightly" packed together in the LLE-based representation 2d than in the PCA-based representation 2c.

5.2 Clustering

Clustering via affinity propagation yielded inadequate results when used on the raw end-game statistics as well as the LLE projected data, producing several small, independent clusters as opposed to larger, more meaningful ones that one would hope for. These results were therefore left out from the previous section for brevity. When used on the PCA projected data however, we find 14 well defined clusters as seen in 3. When compared to the primary champion tags assigned by Riot (as seen in 2c, our method produces further sub-divisions of their groupings when visualized in our data.

5.3 Recommendation

An ideal validation of this paper's recommendation system would involve system testing by a large set of players, tracking each player's winrates given their recommended champions and playstyles while tracking error between the player's functional performance metrics for each of the segments and the recommended functional performance goals given each of the segments. 4a and 4b depict two sample cluster and champion recommendation lists for held out players, demonstrating the types of histograms which might ultimately be compared against similar histograms of wins for novel players who utilizes the system with recommended champions. 4c demonstrates the regression system's recommendations for a player's early-match and mid-match behaviour. For the early-match recommendation, the metrics for a random example from the player's match wins and the recommendation system's error compared to the the example are shown following the goal metrics. The mid-match recommendation additionally shows how the true early-match data was used to recommend the mid-match behaviour. This demonstrates how one might ulimately track error between goals and true statistics for novel matches guided by the recommendation system.

6 Conclusion

Thus far, we have created a system capable of self-clustering playstyles given a pool of match data. It is able to fit a novel players matches into one of these playstyles and make overarching strategic recommendations (such as champion choice), as well as provide in-game recommendations such as goals a player should be looking to achieve by the 10 and 20 minute mark. We believe that such a system could bring about a revolutionary method of self-improvement wherein a player is able to better understand his/her shortcomings and pin-point the particular aspects of the game where he/she might be lacking - whether it be too many deaths, too little champion kills, or too little minion kills. In the end, we hope, that players - both casual and professional are able to use this system and derive benefit from it, from climbing the ladder in ranked matchmaking, to beating out friends, or even becoming the next big player on the scene that rises up above the rest to take home the championship.

7 Future Direction

In the near future, we hope to be able to obtain a developer's API key from Riot. This would raise the limit on the amount of data we are able to request from their servers and give us access to a more continuous stream of match data to work with. In response to this stream of data, we would like to be able to update our dimensionality reduction to instead use an online algorithm such as incremental PCA which, in essence, would give us a recommendation system that is able to stay relevant even when updates are made to the game through different patches. Eventually, we would also like to move our analysis to different games in the same genre such as Dota 2.

References

[1] Thomas E. Batsford. Calculating optimal jungling routes in Dota 2 using neural networks and genetic algorithms. *Game Behaviour Vol 1, No 1, University of Derby*, 2014.

[2] Rohit Bhattacharya and Azwad Sabik. dota2-league-rep-learning. `https://github.com/rbhatta8/dota2-league-rep-learning`, 2015. [Online; accessed 9-December-2015].

[3] Rohit Bhattacharya and Azwad Sabik. Winsight. `http://technical.ly/baltimore/2014/02/04/hophacks-2-johns-hopkins-hackathon/`, 2015. [Online; accessed 9-December-2015].

[4] James Dator. The International Dota 2 championships final results and prize money. *SBNation*, 2015.

[5] Martin Decoud. datdota. `http://www.datdota.com/`, 2015. [Online; accessed 9-December-2015].

[6] David Cournapeau et al. scikit-learn. `http://scikit-learn.org/stable/#`, 2015. [Online; accessed 9-December-2015].

[7] Brendan J. Frey* and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315, 2007.

[8] Ian T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.

[9] Elo Entertainment LLC. Dotabuff. `http://www.dotabuff.com/`, 2015. [Online; accessed 9-December-2015].

[10] Inc Riot Games. Riot Games API. `https://developer.riotgames.com/`, 2015. [Online; accessed 9-December-2015].

[11] Sam Roweis and Lawrence Saul. Nonlinear dimensionality reduction by locally linear. *Science, v.290 no.5500 , Dec.22, 2000. pp.2323–2326.*

[12] AG Stephan. Riot-Watcher. `https://github.com/pseudonym117/Riot-Watcher`, 2014. [Online; accessed 9-December-2015].

[13] Paul Tassi. Riot's 'League of Legends' reveals astonishing 27 million daily players, 67 million monthly. *Forbes*, 2014.

[14] Adrian Touma and Marius Tibeica. Dota picker. `http://dotapicker.com/`, 2015. [Online; accessed 9-December-2015].

[15] Patrick Walker. Dota 2 impresses with super engaged player base - EEDAR. *gamesindustry.biz*, 2015.