MIT SLOAN
SPORTS ANALYTICS CONFERENCE
MARCH 3 - 4, 2017 HYNES CONVENTION CENTER

An Open-Sourced Optical Tracking and Advanced eSports Analytics Platform for League of Legends

Paper Track: Other Sports
Paper ID: 5553

1. Introduction

Being digital, eSports should have vast quantities of data, but sometimes they do not. League of Legends (LoL), the most-played multiplayer online battle arena (MOBA) game, only has boxscore-equivalent statistics provided by the game's publisher (Riot Games) through their free application programming interface (API) [1]: things like when each champion died, or got a kill or an assist, or similar. All current LoL analytics from amateurs to pros rely on this rudimentary API data.

We have developed a new and unique way to capture the far more abundant and useful underlying data. Among other things, we track every champion's location multiple times every second. Physical sports have been revolutionized in the past few years by similar optical tracking data for players (c.f. [2], [3]); what we do for eSports goes even further. In addition to the location data, we also track every ability cast, every attack made, and the damages caused and avoided. We track the amount of actual vision granted and denied in the fog of war. We track every player's health and mana and cooldowns. We track everything, continuously. And we do it invisibly, remotely, and live, all without any impact on the user, or even their knowledge[1].

From this unique and comprehensive data, we are also able to define much better stats and analytics than what is possible from the API. Think of all the things critical to the game that cannot be determined from the API data. When you died, how many allies were around you? How many enemies? Why did you die? Was it for lack of vision? A deficiency in health or gold spent? Outnumbered? Ganked? Or simply mechanically outplayed? Was your death a worthless one, or did your team come out ahead overall? If you killed someone, was it a smart kill, meaning one that increased your team's likelihood of winning, or was it a dumb kill that padded your stats but did not help your team? How much time did you spend doing productive activity versus how much time was wasted? In team fights, were your formations correct, tanks in the front, squishies in the back? Did you peel? Did you engage and disengage correctly? How effectively did you use your summoner spells, and did you use them offensively or defensively? Without our data, there is no systematic way to answer any of these crucial questions, and many others. Yet these are exactly the kinds of questions that LoL players, casual and professional, ask themselves when they review game film.

All our code and data will be open-sourced to make it freely available without restriction to other researchers, analysts, professionals, students, and gamers. Imagine if optical tracking data in other sports were available to everyone for free; that is what we are offering to the community to truly kick-start a revolution in eSports analytics. We began with LoL because it has by far the largest player base [4] but our general methodology is applicable to virtually all eSports. In addition to our code and data, our state-of-the-art analytics platform will also be open-sourced and free.

[1] There are no privacy concerns here: we only track games that allow public spectating.

Furthermore, we have found that the insights from eSports analytics using our advanced data can also inform analytics in traditional sports. In particular, in this paper, we will use our unique data to present a novel approach and answer a question that is of important general interest to *all* sports: how much does an individual's performance contribute to a team's likelihood of winning?

This is a central question of sports analytics for traditional sports as well. Like points scored or defended in traditional team sports, eSports measure kills and deaths. But, again like traditional team sports, these standard metrics do not do a great job of predicting team performance. There is some correlation to be sure, but high scoring players on bad teams are so common as to be a cliché—in all sports. Unfortunately, there are not that many games to analyze in traditional sports. Esports, however, is different; millions of games are played every day.

We tracked and analyzed millions of games of LoL using our new methodology for extracting high-frequency and high-fidelity eSports data. Next, we developed and calibrated a live, in-game win probability model based on the conditions of the game at each moment (see Figure 1). Finally, we improved the standard metrics to look at *worthless* deaths and *smart* kills. Worthless deaths are ones that do not increase your team's win probability. Smart kills are ones that do increase your team's win probability. With these two new metrics, suddenly the relation between individual performance and team performance is virtually a straight line. Figure 2 shows the difference between the upward sloping but relatively flat relationship of the standard stats on the left-hand side, and the practically straight one-to-one relationship of our new advanced stats on the right.

Not only do these results suggest specific new ways forward for traditional sports, they also provide a method for generating brand-new insights: try things on eSports, where the data is far larger and cleaner, and carry those lessons back to the non-e-sports. Finally, with eSports itself poised to take over traditional sports in the future, these insights will be valuable for their own sake as well, in addition to improving all other sports analytics.

Figure 1: Win Probability

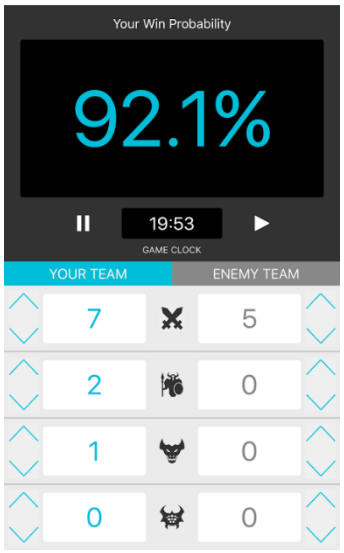
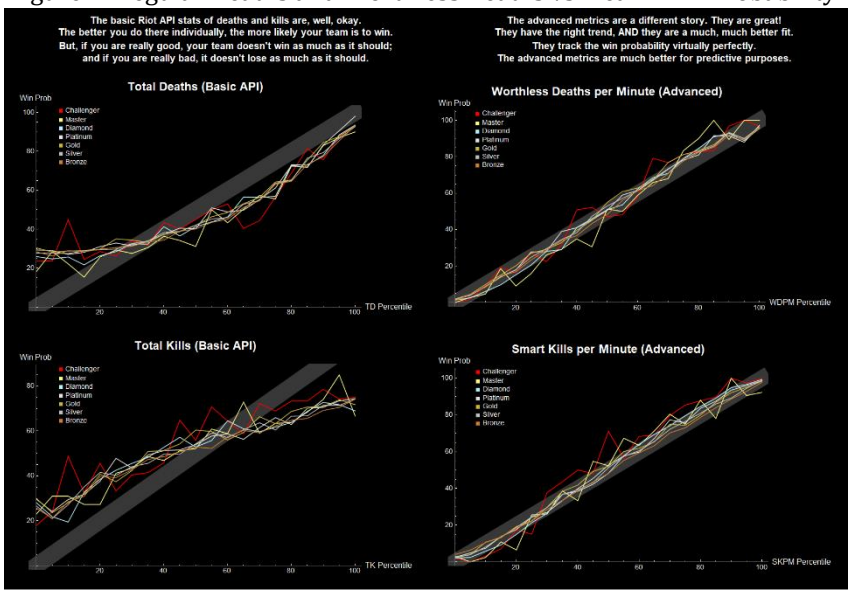


Figure 2: Regular Deaths and Worthless Deaths vs. Team Win Probability



2. Data Methodology

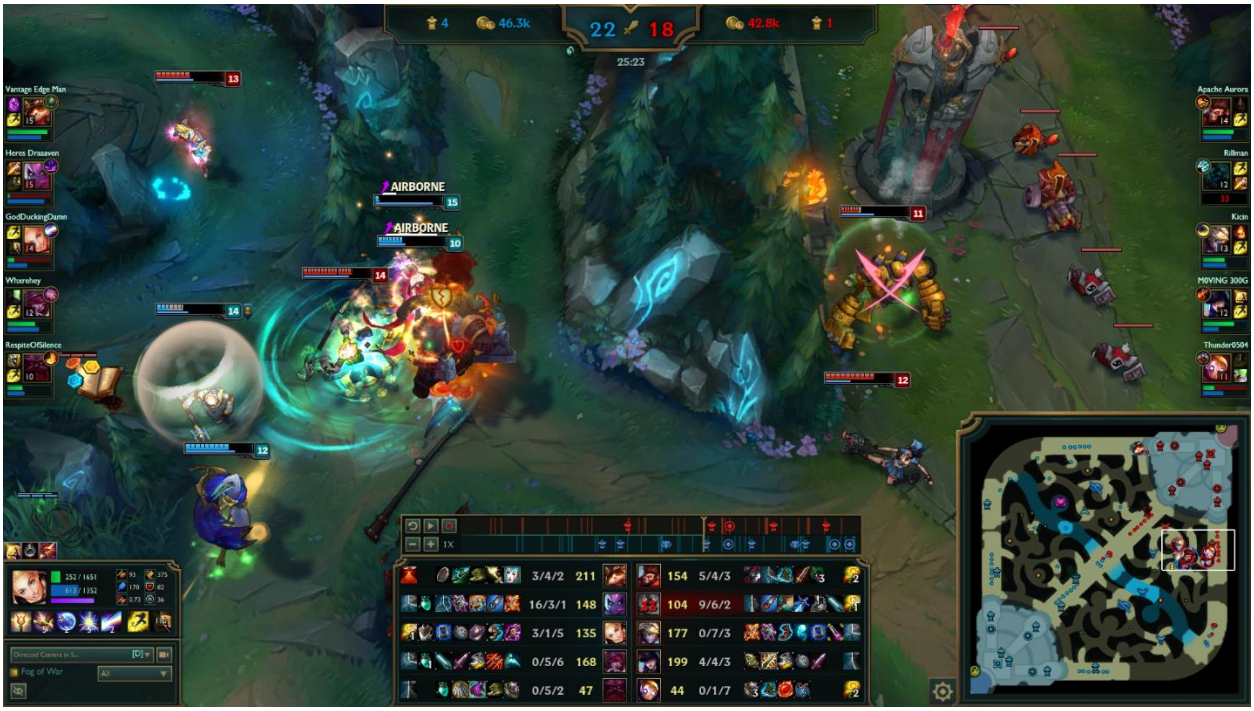
We have developed three independent methods for extracting detailed gameplay data, and have made some progress with a fourth method. This section explains the basic principles behind our approaches in a high-level overview. The full details and all source code for all methods, both front-end and back-end, will soon be made available and open sourced to the analytics community.

When ten humans (referred to as "summoners" to distinguish them from the specific "champions" or avatars they control) play a round of LoL, and allow public spectating of the game, anybody can choose to watch the game live as it unfolds. The normal use cases for this are friends watching each other play, professionals streaming for their coaching staff, and commentators discussing a live professional game. When you spectate someone else's game, you do it through the same game client as if you were playing a game yourself. Rather than being a static video, you have various controls, including camera controls, time controls, and pop up information. You can see both teams if you choose. To avoid cheating, Riot broadcasts spectated games on a three-minute delay. That way, you can't have a friend spectate your game and tell you where the enemy is.

The network architecture is such that a player's actions are sent to Riot's servers, which respond with updates of the game. This happens quickly so that players experience neither a lag between their issuance of a command and the ensuing movement, nor erratic enemy activity. This bidirectional network traffic is highly encoded, difficult to decipher, and subject to change with every patch. Patches happen regularly in LoL, approximately every few weeks.

For our methods of data collection, we rely on the fact that the LoL game client is running and decoding the network traffic appropriately. The results are displayed on the monitor exactly as they would if you were playing the game. Figure 3 shows a snapshot of a spectated game.

Figure 3: A Snapshot of a League of Legends Game in Spectator Mode



Note the plethora of information. While the center of the screen shows the action, and is the primary focus of a live gamer, the top shows the overall progress of each team in terms of gold, turrets, and kills, the bottom shows gold earned, items purchased, and the basic kills/deaths/assists stats, and the left and right champion bubbles show health, mana, level, and important cooldowns, but the most important information is in the bottom right minimap. The minimap shows the locations of all ten champions, turrets, inhibitors, wards, traps, minions, and monsters.

The quintessential approach our eSports optical data tracking software takes is to visually process that minimap and infer all the locations it implicitly shows. We also track the other statistics and information from the other areas, but it is the minimap that is the most challenging and most rewarding. Technically, this approach relies on OpenCV, the industry standard open source computer vision library. Much of the difficulty in processing is due to champion overlaps and various other visually confusing information.

Two of our approaches to data collection rely on such computer vision live tracking using the spectator mode of the game client, one written in Python and one written in Go. A third approach uses direct hooks into the game client to determine actions. And a fourth approach, not yet finished, attempts to interpret the network traffic directly, without the need for the game client itself.

In each case, cloud servers running Windows and game clients are automatically managed, starting and stopping in response to the volume of games requiring analysis. Each new instance launches and prepares the game client, activates spectator mode for the next game in the queue, and begins the optical tracking, shutting down when complete. The cloud instance needs an expensive graphics processing unit (GPU) for the game client to function; hence the motivation for the fourth approach.

Table 1 lists some the data we are now able to extract from games for each champion, each second.

Table 1: Optical Tracking Data for League of Legends

Game State	Minutes Played; Alive/Dead Status and Health of Turrets, Inhibitors, and Monsters
Resources	Champion Health and Alive/Dead Status
	Mana or Other Secondary Resource Bar
	Cooldowns and Status of All Four Abilities and Both Summoner Spells
	Items Owned and Cooldowns if Active
	Ward Cooldowns
	Champion Pings
	Gold Earned and Spent
Statistics	Attack Damage; Ability Power; Movement Speed; Attack Speed
	Cooldown Reduction; Lethality and Armor Penetration; Magic Penetration
	Life Steal; Spell Vamp; Health Regeneration; Mana/Resource Regeneration
	Critical Chance; Critical Damage; Attack Range; Tenacity; Armor; Magic Resist
	Current Buffs such as Baron Buff, Dragon Buffs, or Neutral Monster Buffs
	Current Kills, Deaths, and Assists
Damage	Damage Dealt and To Whom and Type
	Damage Received and From Whom and Type
	Damage Mitigated or Shielded
	Neutral Monster or Minion Damage
	Combos (the Order in Which Abilities and Attacks Were Executed)
Location	X Location and Y Location; Velocity and Acceleration; Stealth Status

3. Analytics Methodology

One analytical advantage of eSports over regular sports is the number of games played. With our data methodology in hand from the previous section, we can let it loose on an ever-expanding set of players, following each of their matches, and adding all that data to our universe. We start with a list of the top LoL players. Players in LoL fall into one of these ranks: Unranked, Bronze 5..Bronze 1, Silver 5..Silver 1, Gold 5..Gold 1, Platinum 5..Platinum 1, Diamond 5..Diamond 1, Master, Challenger. Ranks are determined essentially via an ELO process calculated from your performance, much like a chess rating. By definition, Challenger players are the best 200 players in the world.

Starting with this list of 200, we can spectate all their games. Sometimes their games will include nine other challengers, in which case we simply analyze the game and move on. Sometimes their games will include at least one other non-challenger, so we add that person to our list. And when that person plays games, he likely also plays either with or against other people not already in our corpus, so it continues to grow. By this process, we scraped several million games.

3.1. In-Game Win Probability

Our first important analytics contribution is in calculating in-game win probability using a machine learning model. While there are hundreds of potential metrics each second that could contribute to the win probability (including all the metrics in Table 1), we decided to limit it to the few most salient features so that the win probability could be estimated quickly with a cellphone app by players in-game (see Figure 1). These features are: the red and the blue team's kills, towers, and large monsters killed. We also chose to look at each incremental minute rather than every second.

Our training set thus includes rows like this:

`Minutes Elapsed, Red Kills, Blue Kills, Red Towers, Blue Towers, Red Monsters, Blue Monsters → Win/Loss`

where the Win/Loss indicator output variable is based on the outcome of the complete game.

One immediate issue is that consecutive rows from the same game are highly correlated. To counteract this issue, and because we had the luxury of millions of games, we chose a random minute from each game, leaving us with millions of independent rows.

The machine learning model we used was logistic regression. Following the standard definition and implementation [5], this models the log probabilities of each class (win or lose) with a linear combination of numerical features $x = \{x_1, x_2, \dots, x_n\}$ so that $\log(P(\text{class} = k|x)) \propto x.\theta^{(k)}$ where $\theta^{(k)} = \{\theta_1, \theta_2\}$ corresponds to the parameters for winning ($k = 1$) and losing ($k = 2$). The estimation of the parameters by stochastic gradient descent minimizes the loss function:

$$\sum_{i=1}^{n} -\log(P_\theta(\text{class} = y_i|x_i))$$

Notably absent from our model is the total gold for each team. Both casual observers and professional commentators often use the net gold difference between teams as an indicator of win probability. However, this is an inherently flawed approach, especially as the game progresses, as the team with less gold can often win a team fight, ace the enemies, and march down to destroy the nexus. Furthermore, total enemy gold is not revealed to players in game and must be estimated.

3.2. Full-Game Win Probability

How much does an individual's total performance within a game contribute to his overall team's win probability? A version of this question appears across all sports (c.f. [3], [6]). One benefit of eSports is the vast quantity of games that can be used to answer this question.[2]

Consider a dataset with rows of the following type:

```
Tier, Kills (or Deaths or any other individual metric across the entire game) → Win/Loss
```

where Tier refers to the major portion of the rank described in Section 3, i.e., Bronze, Silver, Gold, Platinum, Diamond, Master, and Challenger, ignoring the 5, 4, 3, 2, or 1 division within a tier.

Within each tier, we can compute the percentiles for the metric in question. For specificity, consider a player's raw number of kills. This kills number will be replaced by its percentile, meaning, he did better than what portion of all players in his tier across all games in our dataset. Thus we now have:

```
Tier, Kills (or other metric) Percentile → Win/Loss
```

Now we accumulate average win/loss percentages within each integer percentile (e.g. 3rd percentile, 27th percentile, and so on) to create a graph of the relationship between individual performance of a single full game metric measured as a percentile relative to that individual's tier, and the team's win/loss probability. This results in graphs such as the left-hand side of Figure 2.

3.3. Worthless Deaths and Smart Kills

Given our in-game win probability model, we can refine the standard metrics of kills and deaths depending on if the team's probability of winning increased or decreased following the event. Specifically, let us call a *worthless death* one that did not increase your team's win probability and a *smart kill* one that did. If you died but took out an enemy inhibitor, that is virtually always worth. But if you chased a weak enemy for two minutes to kill them, that is almost surely not a smart kill, because it was a waste of time and you likely missed out on an objective. With these new metrics, we can repeat the process of Section 3.2 to generate graphs such as the right-hand side of Figure 2.

3.4. Basic and Advanced Stats

Beyond only worthless deaths and smart kills, our new data gives us an opportunity to define dozens of new useful stats. Table 2 lists both the basic stats that can be computed from the Riot API and the new advanced stats. For space considerations, we discuss only the most important.

Some of the stats measure activity. This includes the number of attacks, the number of each of the four abilities used, and the damage per minute caused from combinations of attacks and abilities. Others measure your team fights. A favorable team fight is one where you and your allies outnumber the enemies. How often did such fights happen, and how many kills and deaths resulted? Similarly for unfavorable team fights where you were outnumbered, and balanced fights.

Smart kills and worthless deaths are further broken down depending on *why* the death occurred. Was it due to being in an unfavorable team fight? Or from a lack of vision? Or a large difference in health or gold spent between the champions in the fight? And so on.

[2] Note that our corpus of millions of games spanned several LoL patches, each of which can significantly affect gameplay. Hence, we focused on the ~150,000 games from the latest patch for the purposes of this section, which is still the equivalent of more than a full century of NBA seasons.

Table 2: Basic and Advanced Stats and Bundles

Basic Stats	Advanced Stats
Kills/Deaths/Assists	Damage Taken Percent per Death
CS/Minute Early/Mid/Late/Full	Carry Focus Efficiency
Jungle Minions Killed/Minute	Attacks/Minute
Enemy Jungle Minions Killed/Minute	Ability Counts Q/W/E/R
Gold and Gold Diff Early/Mid/Late	Combo Damage/Minute
Level 6 Seconds and Diff Seconds	Map Coverages Early/Mid/Late/Full
Wards Placed Early/Mid/Late/Full	Useful Percent Early/Mid/Late/All
Wards Killed Early/Mid/Late/Full	Favorable Team Fight Percent
Total Damage Dealt/Minute	Favorable Team Fights Count/Kills/Deaths/Net
Total Damage to Champs/Minute	Balanced Team Fights Count/Kills/Deaths/Net
Total Damage Taken/Minute	Unfavorable Team Fights Count/Kills/Deaths/Net
Total Heal/Minute	Reveals per Ward Average
Average Death Time Percentage Won	Live Wards Average Yellow/Pink/Blue
	Smart Kills/Worthless Deaths Total
	Smart Kills/Worthless Deaths Numbers Difference
	Smart Kills/Worthless Deaths Lacking Vision
Early: Zero to Ten Minutes	Smart Kills/Worthless Deaths Health Difference
Mid: Ten to Twenty Minutes	Smart Kills/Worthless Deaths Gold Spent Difference
Late: Twenty Minutes till End	Smart Kills/Worthless Deaths Neutral Damage Difference
Full/All: Entire Game	Smart Kills/Worthless Deaths Other

Bundle	Component Stats
Abilities	Attacks/Minute
Gold	Gold Early, Gold Diff Early
Laning	Level 6 Seconds, CS/Minute Early, CS/Minute Diff Early
Survivability	Damage Taken Percent per Death
Time Management	Map Coverages Full, Useful Percent All
Carry Focus	Carry Focus Efficiency
Vision Denial	Wards Killed Per Minute Full
Vision	Live Wards Average Yellow/Pink/Blue, Reveals per Ward Average
Deaths	Average Death Time Percentage, Worthless Deaths Total per Minute
Favorable Fights	Favorable Team Fight Percent, Smart Kills Total per Minute

Determining something as fundamental as team fights is a non-trivial problem. First, of course, it requires our new optical tracking data to know the locations of all champions every second. Second, it further requires some judgment and game knowledge to disambiguate team fights. We do this by starting with *engagements*: an engagement is a continuous sequence of champion-on-champion damage until a few seconds elapse with no damage. During engagements, we track the damaged champion and the one who dealt the damage, and include in the team fight any champions within a certain small range of any champion already deemed to be in the engagement. Thus, engagements get concatenated together as champions join or leave a team fight, either by dying or retreating.

4. Results and Discussion

In comparison to other sports for which optical tracking has become available in recent years, the journey from data to actionable results is shorter and better in eSports.

In basketball, for example, optical tracking data consists of player and ball locations only; even basic actions such as dribbles, passes, and shots need to be inferred, which is sometimes a straightforward process, and sometimes not. But the more useful actions, such as picks and screens or cuts or post-up position, etc., are notoriously difficult to determine from the raw location data. Progress has certainly been made, but over many years, requiring many contributions, with complicated machine learning models (c.f. [2], [3]).

In eSports, on the contrary, we are blessed not only with raw location information, but the actions as well. Who attacked who, with what ability, for how much damage; who else was nearby; what buffs, levels, items, and offensive and defensive stats did each champion have; where were wards placed and what vision was available; etc. Thus, with our new methodology, we can get to important and actionable information virtually immediately.

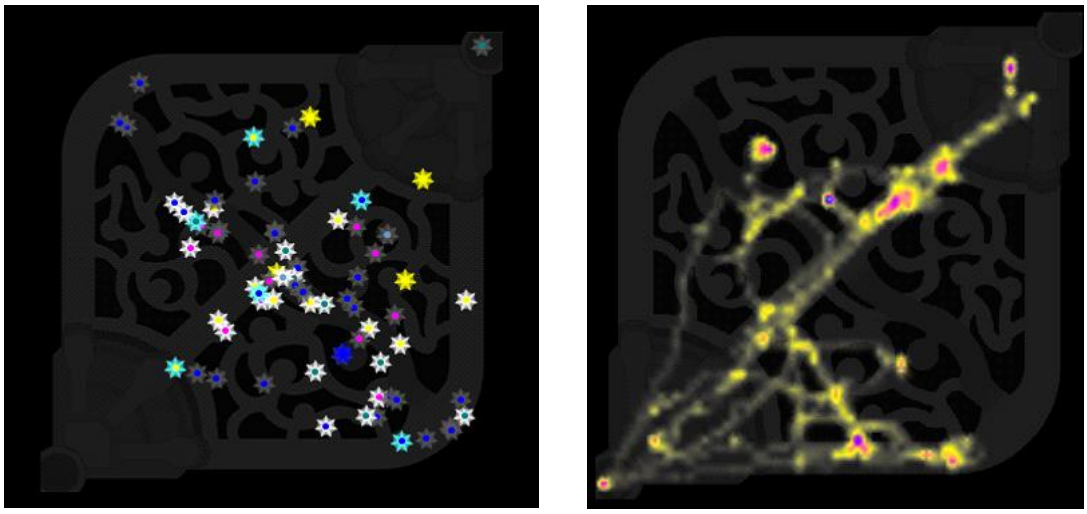
4.1. Examples of Advanced Analytics

Some of the immediately useful information is simply visual presentation of the raw data, while others also require additional computation, such as the in-game win probability. One immediate benefit of our approach is better detection of which champion played which role, a categorization that the official Riot API often fails at. The Appendix shows pseudocode for our improved detection.

4.1.1. Ward Maps and Heat Maps

As a visualization example, Figure 4 shows a ward map and a heat map, neither of which are possible with API data. The ward map shows where wards were placed to provide vision, coded to indicate both the type of ward and its ultimate fate: whether it expired by itself, or was replaced or destroyed, or persisted until the end of the game. The heat map indicates the main locations a specific player occupied on the map. In the case shown, it was clearly a player in the mid lane, who roamed bottom a few times, and to the baron pit, before ultimately pushing through to the nexus.

Figure 4: Ward Map (left) and Heat Map (right)



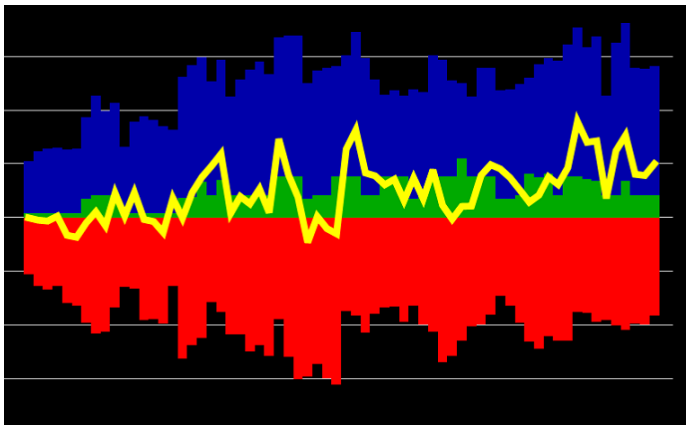
4.1.2. Vision

Beyond simply counting or plotting wards, we can also compute the union of all the regions of vision granted by still-living wards using their specific radius at each moment in time for each team.

Figure 5 shows this visualization, with your team's vision as a percentage of the total map on top in blue, the enemy team's vision on the bottom in red, your team's net vision as the yellow line, and your personal vision contribution in the middle in green.

In this specific game, your team slowly but surely built a vision lead. Your contribution was crucial.

Figure 5: Map Vision Analysis

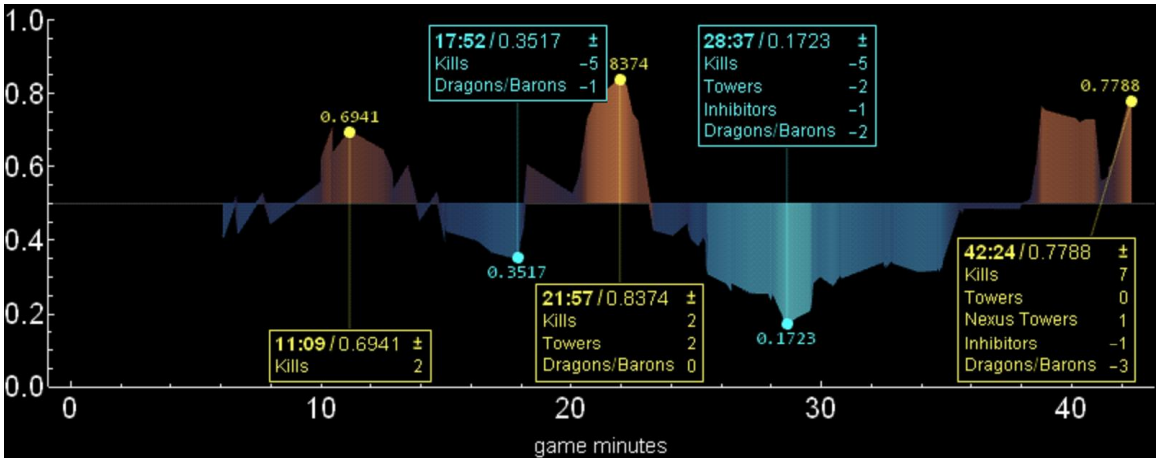


4.1.3. Post-Game Win Probability with Automatic Annotations

It is common in many sports to see graphs of win probability after a game. The large fluctuations especially at ends of games can be dramatic. We further extend this standard type of graph with automatic annotations indicating key states of the game at local minima and maxima.

Figure 6 shows an example. Your team got a few early kills (69%), but then had seven net deaths and a lost dragon (35%), a brief mid game bump followed by a long slump of lost towers and lives (17%), before finally rallying with a sequence of good kills and likely an ace.

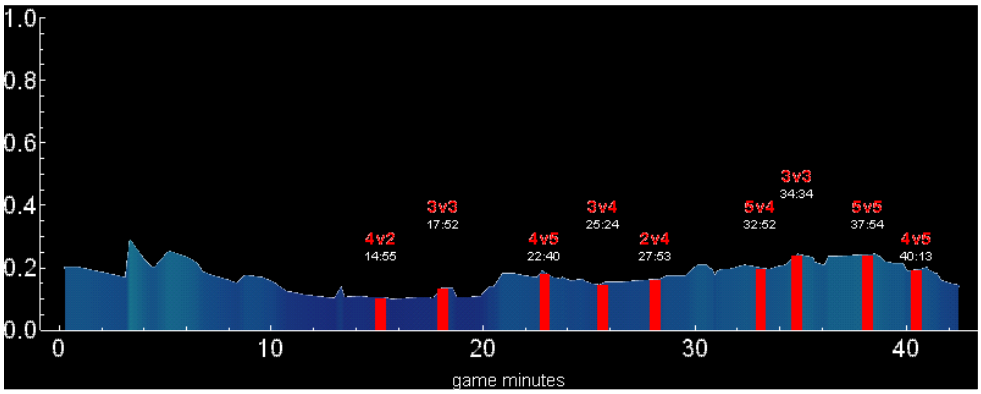
Figure 6: Enhanced Post-Game Win Probability Analysis



4.1.4. Tilt Analysis

A common psychological issue in LoL is the concept of tilt. Borrowing the term from poker, tilt in LoL refers to situations where a player plays suboptimally because of emotional turmoil. The most extreme form of tilt occurs when a player dies in lane then returns immediately and plays too aggressively and dies again and again. We can proxy for tilt by calculating the time spent dead by the player over the preceding three minutes and dividing it by the total time spent dead by anyone on his team over the same period. This can be calculated from basic Riot API data, and produces the blue chart shown below in Figure 7. However, this calculation by itself provides little context or explanation. If we further highlight all the player's deaths in red, and note the kinds of team fights they were involved in (which requires the new data), then the context helps to understand and possibly alleviate future examples of tilt. In the example below, notice the player first died in a favorable and then a balanced team fight, before taking three unfavorable fights in a row.

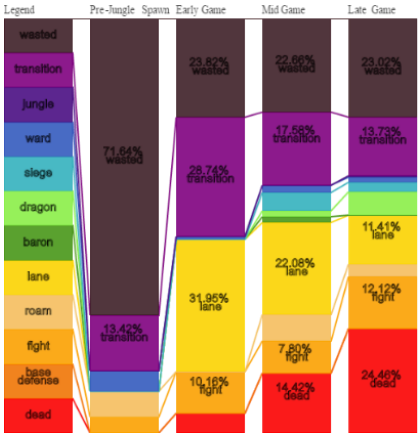
Figure 7: Tilt Analysis and Engagements



4.1.5. Time Management

Given the location and damage target information, we can partition each player's activity at any given moment into one of several types. An example of a post-game time management analysis is shown below in Figure 8. As is common, before minions spawn, most time is essentially wasted. Reducing the amount of wasted time during the rest of the game, however, is an important metric. Transition time, while necessary, also ought to be minimized. You should also expect your laning time to decrease as the game progresses, with the possible exception of split pushers.

Figure 8: Time Management

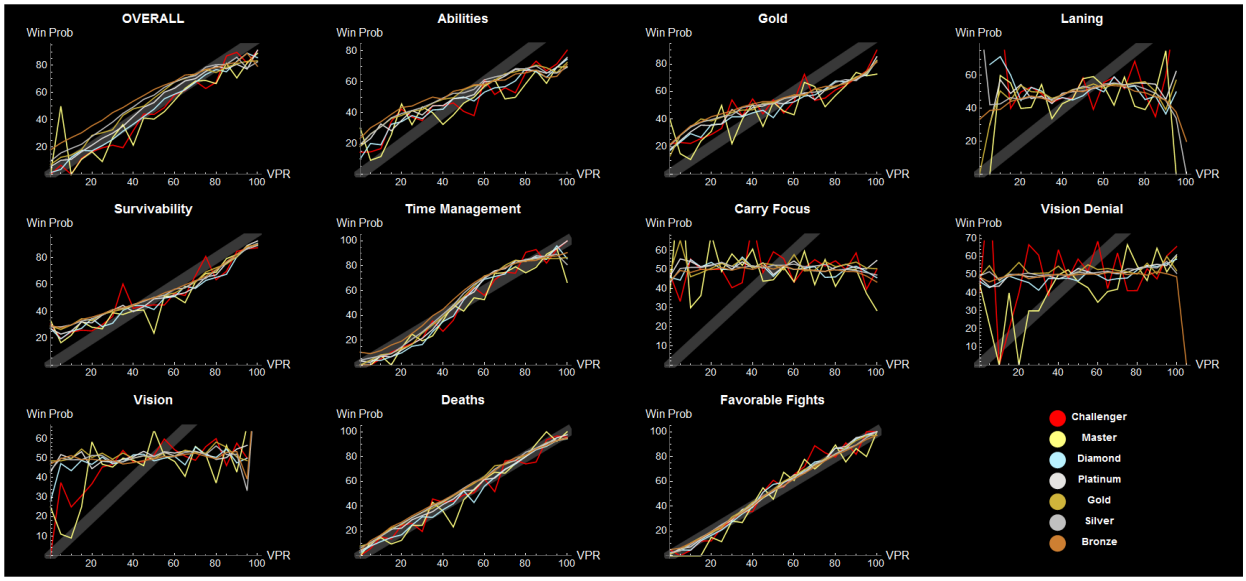


4.2. Individual Performance and Team Win Probability

How does an individual's performance within a game correlate to his team's probability of winning? We previewed earlier in Figure 2 that the relationship between standard kills or deaths with winning is positive but imperfect, similar to the relationship one might expect between, say, an individual's scoring and his team's winning in a traditional sport such as basketball. Filtering individual production for activities that benefit the team, however, has a strong effect.

Figure 9 shows the win probability relative to the percentile for various non-overlapping bundles of advanced stats, bundled as shown in the bottom of Table 2, and separated by tier. The OVERALL figure is for the average of all the bundles, as it is a proxy for an individual's overall in-game performance. Averaged across games, it can also represent a player's overall individual rating.

Figure 9: Individual Performance in a Bundle of Stats vs. Team Win Probability



The OVERALL metric does well. It satisfies the main criteria of being a good metric, namely, as your percentile performance on the metric in a game increases, your team's overall win probability also increases, for each tier. However, it is not optimal because it includes several bundles that do very poorly at explaining a team's win probability: laning, carry focus, vision denial, and vision. An individual's performance in these four bundles seems to be irrelevant to a team's win probability. An OVERALL metric ignoring these four bundles would do even better.

Some of the bundles that do well are intuitively reasonable—deaths and favorable fights line up perfectly and make sense: don't die, and don't take bad fights. Gold also is increasing but not as much. As we know, gold is good but it is not the whole story. Similarly for abilities and survivability.

Time management is an intriguing one, as it measures an individual's impact across space and time, through map usage and portion of time spent in useful activities. To the extent you had one of the best games in terms of spacetime presence for your rank, you are virtually guaranteed to win.

These graphs, however, only show the isolated effects of each bundle separately. We can further consider the marginal contributions in a complete model with a logistic regression.

4.2.1. Logistic Regression

We fit a standard logit model on the ten stat bundles percentiles and a constant term. The estimates β_i and their corresponding z-statistics in the logistic model $\hat{y} = 1/(1 + e^{-(\beta_0+\beta_1 f_1+\beta_2 f_2+\cdots)})$ are shown in Table 3 below, for the corresponding f_i that map onto the ten bundles listed.

Table 3: Team Win Probability Regressed on Stat Bundles

	Estimate	**z-Statistic**
(constant)	0.2132***	4.42
Favorable Fights	0.0378***	99.90
Deaths	0.0595***	127.85
Vision	-0.0088***	-23.63
Vision Denial	-0.0018***	-8.26
Carry Focus	-0.0035***	-13.70
Time Management	0.0499***	102.87
Survivability	-0.0229***	-55.90
Laning	-0.0023***	-4.68
Gold	0.0058***	19.30
Abilities	0.0007*	2.21

* significant at the $p < 0.05$ level
*** significant at the $p < 0.0001$ level

All the estimates are highly significant at the $p < 0.0001$ level except for Abilities, which has a p-value of 0.03. As noted in footnote 2, our sample size was approximately 150,000 games.

Surprisingly, several of the bundles have negative marginal contributions, including survivability. Conditional on all the other bundles, it is actually better for you to have less survivability. Similarly with vision, vision denial, carry focus, and laning. There are two legitimate possible explanations for these negative marginal stat bundles. One is that those metrics tend to correspond to a measure of passivity rather than aggressiveness, and aggressiveness tends to win games even if on the margin it makes those stat bundles lower. The other possibility is of multicollinearity among the features.

Table 4: Pairwise Correlations Among the Features

	Fav. Fights	Deaths	Vision	Vision Denial	Carry Focus	Time Mgmt	Surviv-ability	Laning	Gold
Deaths	0.28								
Vision	0.12	0.28							
Vision Denial	0.23	0.20	0.03						
Carry Focus	0.51	0.32	0.15	0.36					
Time Management	0.10	0.12	0.04	0.04	0.10				
Survivability	0.06	0.04	0.04	0.09	0.14	0.01			
Laning	0.05	0.09	0.05	0.08	0.20	0.02	0.26		
Gold	-0.21	-0.19	-0.03	-0.75	-0.28	0.04	-0.03	-0.01	
Abilities	0.20	0.29	0.02	0.27	0.45	0.08	0.05	0.06	-0.29

Table 4 shows the pairwise correlations of each of the features. Very few exhibit substantial multicollinearity. The biggest positive correlation is 0.51 between carry focus and favorable fights. This makes sense because someone who is able in team fights to focus the carry (defined as the enemy's largest damage dealer) is likely also able to avoid fighting in unfavorable conditions. The biggest negative correlation is -0.75 between vision denial and gold. This also makes sense because support players tend to sweep for enemy wards, and they usually earn relatively little gold.

With roles in mind, we can redo the logistic regressions for each of the five standard roles: top lane, jungle, mid lane, attack damage carry ("adc"), and support. Table 5 lists the z-Statistic for the relevant coefficient in a logistic regression restricted to the given role.

Table 5: Role-Specific Logistic Regression z-Statistics

	Top	**Jungle**	**Mid**	**ADC**	**Support**
(constant)	-2	+4	-2	-6	+16
Favorable Fights	+42	+38	+48	+50	+43
Deaths	+58	+57	+54	+52	+61
Vision	-10	-10	-11	-10	-12
Vision Denial	-6	-3	-4	-4	-2
Carry Focus	-2	-4	-8	-9	-8
Time Management	+47	+52	+42	+46	+45
Survivability	-24	-28	-24	-23	-29
Laning	+2	-5	+1	-1	-6
Gold	+6	+7	+12	+7	+3
Abilities	-2	+4	-2	-6	+16

Certain bundles are highly significant across all roles: favorable fights, deaths, vision, time management, and survivability. Carry focus is significant for all roles except the top lane; the top laner often split pushes and does not join team fights, so he ends up focusing whoever comes to stop him rather than the enemy carry. Laning is important for laners (top, mid, and adc), and not important for junglers and supports, while abilities are the opposite.

For completeness, we can see the separate correlations of each bundle with win probability, by role. Table 6 shows that the top factors are as expected and relatively constant across roles. The implied advice to be a better player is: fight good fights, don't die, and don't waste time.

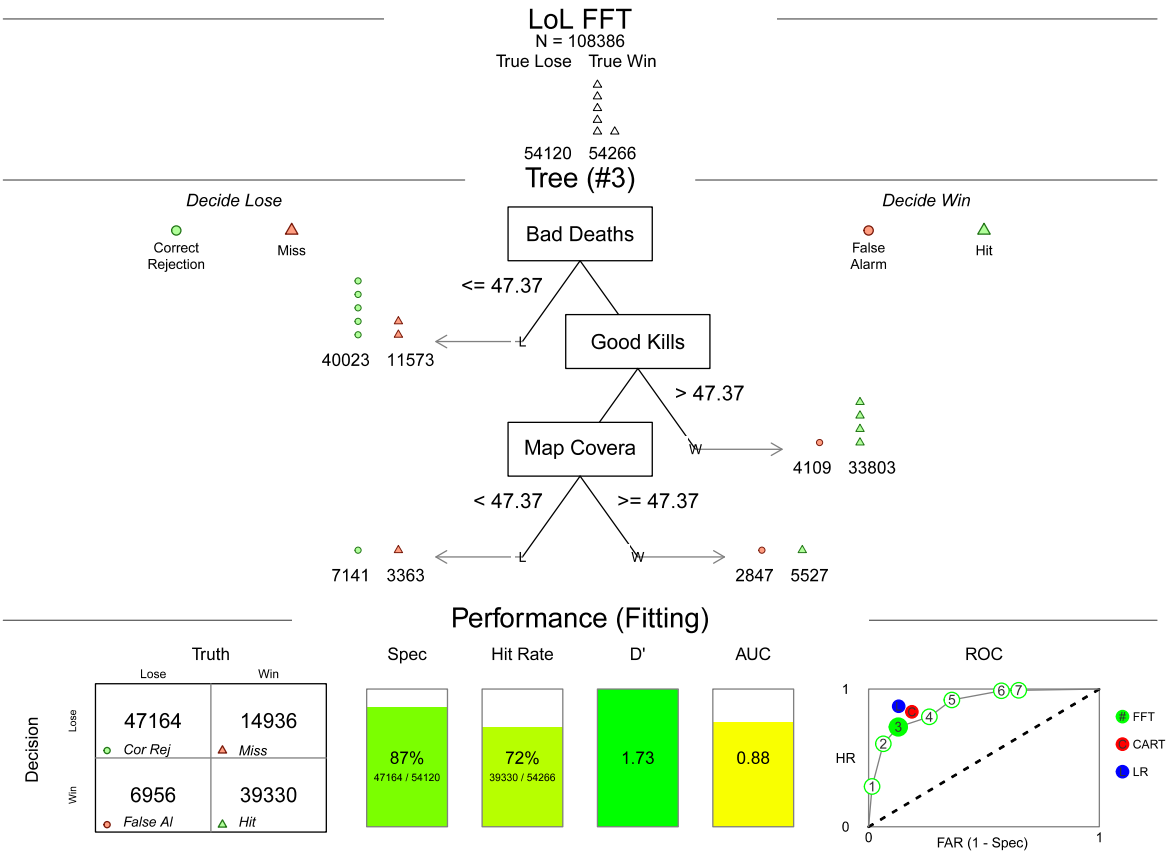
Table 6: Role-Specific Isolated Correlations of Stat Bundles with Win Probability

	Top	**Jungle**	**Mid**	**ADC**	**Support**
Favorable Fights	0.4347	0.5067	0.4890	0.5015	0.5007
Deaths	0.4858	0.5474	0.4970	0.4907	0.4976
Vision	0.0499	-0.0082	0.0218	0.0028	0.0177
Vision Denial	0.0251	0.0092	0.0224	0.0297	0.0394
Carry Focus	0.0254	-0.0178	0.0186	0.0595	-0.0608
Time Management	0.4850	0.4929	0.4875	0.5396	0.4565
Survivability	0.3447	0.3428	0.3489	0.3796	0.3360
Laning	0.1264	-0.0514	0.1338	0.1246	-0.0459
Gold	0.2395	0.2615	0.2652	0.2802	0.2396
Abilities	0.2347	0.3011	0.2556	0.3170	0.2744

4.2.2. Fast and Frugal Trees

We compute a fast and frugal tree to see the best way to predict your team's performance given only the most important details about an individual's performance. Fast and frugal trees (described in general as well as the implementation used here in [8]) are decision rules that are both easy-to-use and make good predictions. We train on all the metrics of Table 2, with 25 percent of the data, about 35,000 observations, reserved for cross-validation. Figure 10 shows the best resulting tree. It makes the correct prediction about 80 percent of the time despite only using three metrics.

Figure 10: Fast and Frugal Tree Analysis for Individual Percentile Performance to Predict Team Victory



The final decision rule is simple indeed: as a rule of thumb, if your bad death percentile was below average (meaning you had more worthless deaths than players at your rank typically do in games), then you likely lost. If, on the other hand, you didn't die that much, then, if you had more smart kills than your peers, you likely won. Finally, if you had both above average worthless deaths and below average smart kills, then whether your team won or not depends on how much influence you had over the map. If you had more map coverage then average, you probably won; otherwise, you probably lost. Thus, the fast and frugal tree (FFT) clarifies our story: don't die for no reason; if you do, try for some smart kills; if you can't do that either, at least roam and have decent map coverage.

Note also that the FFT was free to choose any decision tree using any of the metrics in Table 2, including both the basic and advanced stats, and it ultimately chose to use only the advanced stats, and exactly the ones we had identified in the previous section as having a consistent impact on win probability across all roles. Note also that the FFT correctly predicts victory 87 percent of the time.

5. Conclusion

Using a combination of computer vision, dynamic client hooks, machine learning, visualization, logistic regression, large-scale cloud computing, and fast and frugal trees, we generated new and unique data on millions of League of Legends games, calibrated a win probability model, developed enhanced definitions for standard metrics, presented automated improvement analysis, provided a framework for determining an individual's contribution to a team's victory, and applied that framework on the basic and our new advanced stats to show that the advanced stats both better correlate with and explain team outcomes.

How much does an individual contribute to his team's success? This may be one of the touchstone questions of sports analytics for all sports. Here, we find that standard metrics of performance are insufficient to explain team success in eSports. We also find that an intelligent adjustment to those standard metrics does indeed explain team success. For League of Legends, that adjustment required a substantial investment in original data collection, storage, and analysis across millions of games, but the final insight is quite intuitive: when standard individual actions of performance within a game are filtered for broader impact, the new adjusted summary statistics correlate almost perfectly to team performance. In other words, in the parlance of more traditional sports, one athlete's points and assists do not necessarily correlate with team performance. Indeed, it can often be negatively correlated. However, if we use the insights from here, we can consider *smart* points and assists, ones that not only padded the individual's statistics, but also significantly contributed in the moment to an increase in the win probability of the team.

The code for our entire process will soon be open sourced. This will include the code for extracting the unique data from spectated games, the code for all our analytical models, and the code for generating the types of graphs, charts, tables, and results shown in this paper. We are excited to see what further amazing analytics will be created over the next decade from this release.

Acknowledgments

We are grateful to our professional eSports team partners who have used our data and analytics in their own processes. And we thank (REDACTED FOR ANONYMITY) for their contributions.

References

[1] Riot Games. "Riot Developer Portal." Retrieved from https://developer.riotgames.com.
[2] Maheswaran, Rajiv, et al. (2012) "Deconstructing the rebound with optical tracking data." *MIT Sloan Sports Analytics Conference*.
[3] Cervone, Dan, et al. (2014) "POINTWISE: Predicting points and valuing decisions in real time with NBA optical tracking data." *MIT Sloan Sports Analytics Conference*.
[4] Paul, Jeremiah (2017) "By the Numbers: Most Popular Online Games Right Now." Retrieved from https://nowloading.co/posts/3916216.
[5] Wolfram Research (2017) "Logistic Regression (Machine Learning Method)." Retrieved from http://reference.wolfram.com/language/ref/method/LogisticRegression.html.
[6] Duch, Jordi, et al. (2010) "Quantifying the Performance of Individual Players in a Team Activity." *PLoS ONE* 5(6): e10937.
[7] Phillips, Nathaniel D. (2016) "Making fast, good decisions with the FFTrees R package." Retrieved from http://nathanieldphillips.com/2016/08/making-fast-good-decisions-with-the-fftrees-r-package.

Appendix

The below presents pseudocode for determining the role of each champion, for each team. League of Legends is a flexible game and any champion can in principle go anywhere on the map. However, over time, certain kinds of roles have evolved and become relatively fixed in the community: a top laner, a mid laner, a jungler, and two bottom laners, one of whom supports the other (called the attack-damage carry, or ADC, or simply carry). These roles primarily refer to the laning phase of the game, after which champions frequently roam or group in other parts of the map.

The Riot API reports what Riot assumes are the roles of each person. Their results are sometimes incorrect, particularly if a jungler forgot to bring the Smite summoner spell, or if the bottom champions switched with the top laner for the first few minutes, or if players assigned roles during the game setup phase agreed to swap but then roamed the wrong lanes, or other unusual but still occasionally occurring cases. Using our new data, we have found the following algorithm to work in all cases that we have seen, from amateur play to professional play.

Pseudocode: Automatic Categorization of Champion Roles

```
Early Minions: Define early minions killed as those minions
killed by a champion who at the time of killing the minion was
between level two and level six.

For each team:

The jungler is the champion who killed the most early minions
that were neutral jungle minions.

Of the remaining four champions: the support is the one who
killed the fewest overall early minions.

Of the remaining three champions: the mid is the one who killed
the most early minions near the map's center.

Of the remaining two champions: the ADC is the one who was most
frequently near the support during the first ten minutes of
gameplay, where near is defined as being within one-tenth of the
width or height of the map.

The remaining champion is the top.
```