WIRTSCHAFTSUNIVERSITÄT WIEN
Vienna University of Economics and Business

Master's Thesis

Title of Master's Thesis:	Impacting the Economic Success of Software Products by leveraging Insights from Reddit Communities through Text Mining and Sentiment Analysis for Data-Driven Software Updates - A League of Legends Case Study
Author (last name, first name):	Gulliver Wutz
Student ID number:	01553720
Degree program:	Master in Digital Economy
Examiner (degree, first name, last name):	ao.Univ.Prof. Dr. Johann Mitlöhner

I hereby declare that:

1. I have written this master's thesis myself, independently and without the aid of unfair or unauthorized resources. Whenever content has been taken directly or indirectly from other sources, this has been indicated and the source referenced. I am familiar with the regulations specified in the Directive on Plagiarism and Other Types of Academic Fraud in Academic Theses.

2. This master's thesis has not been previously presented as an examination paper in this or any other form in Austria or abroad*.

3. This master's thesis is identical with the thesis assessed by the examiner.

4. (only applicable if the thesis was written by more than one author): this master's thesis was written together with

   The individual contributions of each writer as well as the co-written passages have been indicated.

18/09/2024
Date

Signature

*This does not apply to master's theses written as part of WU cooperation programs (joint or double degrees).

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

Master Thesis

Impacting the Economic Success of Software Products by leveraging Insights from Reddit Communities through Text Mining and Sentiment Analysis for Data-Driven Software Updates - A League of Legends Case Study

Gulliver Wutz

Date of Birth: 12.03.1996
Student ID: 01553720

Subject Area: Digital Economy

Studienkennzahl: 066 960

Supervisor: ao.Univ.Prof. Dr. Johann Mitlöhner

Date of Submission: 18.09.2024

Institute for Data, Process and Knowledge Management, Vienna University of Economics and Business, Welthandelsplatz 1, 1020 Vienna, Austria

DEPARTMENT FÜR WIRTSCHAFTSINFORMATIK UND OPERATIONS MANAGEMENT DEPARTMENT OF INFORMATION SYSTEMS AND OPERATIONS MANAGEMENT

Contents

List of Figures

List of Tables

Abstract

This thesis explores the critical role of community sentiment and user feedback in driving the economic success of software products, using the game League of Legends as a case study. By mining textual data from Reddit communities and applying sentiment analysis, this approach demonstrates how developers can use unfiltered user generated content to inform software updates and future development decisions. Further, the importance of responding swiftly and effectively to community feedback is highlighted, particularly in the fast-paced and highly competitive video game industry. The insights gained from Reddit reveal key areas of user satisfaction and dissatisfaction, providing developers with a nuanced understanding of how different segments of their player base react to updates. This data-driven approach offers clear economic advantages: products that continuously engage users by addressing community concerns are more likely to maintain loyalty, driving sustained revenue growth in subscription-based or microtransaction-based models. The case study on League of Legends illustrates the broader applicability of this methodology across the software industry, especially in sectors reliant on frequent updates and evolving user needs. Ultimately, it is argued that leveraging community sentiment through platforms like Reddit is not only a powerful tool for improving software quality but also an important strategy for maintaining market competitiveness and long-term profitability.

1 Introduction

In the increasingly competitive landscape of the software and video game industry, the ability to maintain user engagement and satisfaction is an important determinant of a product's long-term success [56][6][8]. This is particularly true for games, where user experience drives both player retention and monetization [36]. As the gaming industry has evolved, so too have the ways in which companies approach software development, updates, and community interaction. Developers and publishers are therefore aiming to extend traditional methods of gathering user feedback such as surveys, focus groups, or beta testing. In an era defined by rapid digitalization and social media proliferation, users now have a multitude of platforms to voice their opinions, share experiences, and critique the products they engage with. The rise of social media has created new channels for feedback, where users can engage with developers and other players alike. Platforms like Twitter, YouTube, and Reddit have emerged as important hubs for community-driven feedback, discussions, and sentiment. These platforms generate large amounts of data daily, offering developers sources of insights that can potentially shape future product updates and business strategies. For software companies, particularly those in the video game industry, leveraging user-generated content from these platforms presents both an opportunity and a challenge.

Over the last two decades, the software and video game industries have undergone a profound transformation in how they deliver content to users. In the early days of software development, products were released as complete, standalone entities, with little to no post-launch updates aside from bug fixes or minor patches. The increased access to the internet, however, introduced the era of downloadable content (DLC), allowing developers to continuously iterate on their software long after their initial release. This shift gave birth to the concept of "software as a service" (as opposed to "software as a product") where regular updates are rolled out to sustain interest, introduce new content, and fix emerging issues [91]. For online multiplayer games in particular, this continuous cycle of updates is important. Games such as "League of Legends", "World of Warcraft", "Fortnite", and many others rely on these regular updates to keep their player base engaged [56]. These updates may include new content such as maps, characters, and storylines, as well as balance changes to address competitive fairness. Yet, one of the biggest challenges developers face in this iterative development cycle is determining what users actually want in terms of which issues need immediate attention, and how to balance the needs of the competitive and casual gaming communities. In a world where user expectations are frequently changing and competition

is strong, maintaining a dynamic and responsive update cycle can make or break a software's success. Therefore, feedback from the community is no longer just a helpful tool. It becomes a critical component in the decision-making process [32]. This is where alternative data sources, like social media platforms, come into play.

Social media platforms may advance how developers gather insights into user experiences. With the rise of platforms such as Reddit, players now have numerous channels to discuss their opinions and experiences with a video game or software product. These platforms serve as informal but highly active forums where players can directly engage with developers, voice their frustrations, offer suggestions, and even form community-driven solutions to in-game problems [22]. In this context, developers are confronted with two challenges: they have access to an immense volume of user-generated feedback, but they must also navigate the challenge of extracting meaningful and actionable insights from this data. Reddit, in particular, stands out as a unique platform due to its structure and community-driven nature. Topic-specific forums moderated by users offer a decentralized space where individuals can engage in deep discussions about specific games, software, or broader industry trends. With its upvoting and downvoting system, Reddit allows the most popular, relevant, or insightful discussions to rise to the top [4], providing developers with a snapshot of the community's most pressing concerns and topics of interest. However, unlike traditional forms of feedback (such as surveys), the data from Reddit is unfiltered and unstructured. Extracting useful insights from this data requires sophisticated techniques such as text mining and sentiment analysis, which enable developers to navigate through large amounts of content and to quantify user sentiment.

Text mining and sentiment analysis have emerged as powerful tools for making sense of unstructured data from social media platforms. Text mining refers to the process of extracting useful information from large volumes of text [3], while sentiment analysis focuses on understanding the emotional tone behind the content [90]. Together, these methods allow developers to analyze user opinions at scale, identify key topics of discussion, and track sentiment over time. This can be particularly useful in assessing how different updates or changes to a game or software have been received by the user base. For developers, text mining and sentiment analysis offer a data-driven approach to decision-making. Rather than relying on isolated feedback from a small group of users, developers can tap into the collective voice of the community to make more informed decisions about future updates. By analyzing trends in community sentiment, developers can prioritize updates

that will have the greatest positive impact on the user experience, while also identifying potential problem points before they escalate into larger issues. For instance, in the game of League of Legends, a developer might use sentiment analysis to track how the community is reacting to a newly introduced character or a balance change to an existing character. If negative sentiment begins to dominate the conversation, the developer can investigate the specific concerns being raised and decide whether a hotfix patch or further adjustments are necessary. Similarly, text mining can reveal emerging topics of discussion, such as requests for new features or complaints about bugs, that developers may not have considered.

The benefits of leveraging social media data for game development go beyond improving player satisfaction. They can also have a direct impact on a company's economic success. In the competitive gaming market, where players have a wide variety of alternatives, player retention is important [56]. Regular updates that address community concerns and introduce new content can keep players engaged and invested in a game over the long term. This, in turn, can drive revenue through microtransactions, expansions, or in-game purchases, which have become a key revenue model for many online multiplayer games [16]. A game that fails to keep up with its player base's expectations risks losing players to competitors, leading to decreased engagement and revenue. On the other hand, a game that actively responds to player feedback and continuously evolves with its community can build a loyal, long-term player base, generating consistent revenue over time. Many online multiplayer games have successfully adopted this model of offering frequent updates while introducing new skins, items, and game passes that keep players financially invested in the game [7]. Thus, there is a clear link between data-driven updates and a game's economic success. By leveraging insights from social media platforms, developers can not only enhance the player experience but also drive revenue growth by keeping their games relevant and exciting for the community.

League of Legends, developed by Riot Games, serves as a prime example of a "game as a service", that has successfully navigated the challenges of community feedback and regular updates. Since its release in 2009, League of Legends has maintained its position as one of the most popular games in the Multiplayer Online Battle Arena (MOBA) genre, attracting millions of players worldwide [28]. The game operates in distinct yearly cycles called "seasons", with each season representing a competitive period during which major updates are introduced. At the time of writing, League of Legends has completed 13 seasons, and is currently running through season 14. The

game's longevity and success can be attributed, in part, to Riot Games' commitment to continuously evolve the game through regular patches and updates. However, with such a large and diverse player base, Riot Games faces the ongoing challenge of keeping all segments of its community satisfied. Competitive players, casual players, and newcomers all have different expectations and desires for the game, making it difficult to strike the right balance. Riot Games has historically relied on traditional feedback mechanisms such as player surveys, forums, and beta tests to gather insights into player sentiment. But in recent years, the company has increasingly turned to platforms like Reddit, where the League of Legends community is particularly active, to gather more informal and spontaneous feedback from players [88].

By analyzing the vast amount of player-generated content on Reddit through text mining and sentiment analysis, the developers have an opportunity to gain a deeper understanding of the community's sentiments and identify recurring topics of discussion. This allows the company to make more informed decisions about future updates, balancing changes, and new content releases. In this thesis, League of Legends serves as a case study subject to explore how insights from social media platforms (specifically Reddit) can be leveraged through text mining and sentiment analysis to inform data-driven software updates. The aim is to demonstrate the broader applicability of this methodology for other games and software products, highlighting how developers can use social media feedback to enhance user satisfaction and drive economic success.

1.1 Research Question

The research approach presented throughout this thesis seeks to answer the following research question: "How can text mining and sentiment analysis of textual Reddit data be used to inform data-driven game updates for League of Legends, with the aim of enhancing player satisfaction and thereby driving economic success?" More specifically, the aim is to explore the potential of using user discussions on Reddit to identify trends in player sentiment regarding specific game elements (such as champions, items, or mechanics).

1.2 Research Method

To address the research question, the methodology presented within this thesis employs a mixed-methods approach, combining both qualitative and

quantitative techniques to analyze Reddit data. The primary method involves text mining and sentiment analysis on a large dataset of Reddit posts from a selection of League of Legends subreddits. Data is collected via the Reddit API to capture relevant discussions over a specified period, particularly focusing on posts and comments related to recent game patches. A combination of TF-IDF term clustering, custom vocabulary dictionaries, and sentiment classification models is applied to uncover patterns in player sentiment regarding game updates. Additionally, the findings from Reddit are cross-referenced with aggregated performance data from platforms such as LoLalytics, METAsrc, and U.GG to assess how online discussions align with actual gameplay trends. This data-driven approach illustrates how Reddit discussions can be used to guide game updates, with the ultimate goal of improving both player retention and the economic performance of League of Legends.

2 Overview of League of Legends

The following subsections provide an in-depth look at League of Legends gameplay mechanics, map structure, game objectives, and competitive ranked mode. The purpose is to show the strategic complexity and dynamic nature of the game. Furthermore, game aspects such as champion roles, in-game objectives, and the skill-based matchmaking system that underpins the highly competitive environment in League of Legends are described.

2.1 Game Genre

League of Legends (LoL) is a PC-based game [22] that is published by Riot Games [30] and belongs to the genre known as MOBA (Multiplayer Online Battle Arena [94]). This genre blends elements from role-playing games, real-time strategy games, and tower defense games [22].

League of Legends was released in late 2009 [52] and has since accumulated an enormous player base. As of March 2023, the game has over 150 million registered users, with approximately 125 million of them being active monthly players. The demographic distribution reveals that males account for 82% of the League of Legends player base. On average, the number of concurrent players exceeds 700,000. The game's popularity extends beyond just its gameplay, as evidenced by audiences watching over 138 million hours of League of Legends on the streaming platform Twitch [101] in August 2021. Looking at its economic success, the game generated around $1.75 billion in

revenue in 2020 (see section 4.4.1 for details on revenue generation). Additionally, the prize pool for the League of Legends World Championship in 2020 was $2.34 million which also highlights the game's significance to the esports industry. [28]

A typical League of Legends match involves two teams, each comprising of five players. Each player controls one of the currently 168 [76] available "champions". These battles are strategic and often hectic, with games usually lasting around 30 minutes, though some matches can extend beyond an hour [22]. League of Legends operates as a server-based game [80], necessitating a continuous connection to a Riot Games server during gameplay. These servers maintain real-time information about the game's current status, ensuring seamless interaction and updates for all players.

2.2 Map and Game Setup

The main gameplay mode of League of Legends is played in a virtual arena called "The Summoner's Rift" (seen in Figure 1). It represents a square-shaped map bisected by a river running diagonally across it. This river divides the bottom-left "Blue" side from the upper-right "Red" side. Each team starts the game at their respective spawn point, or "nexus", located in these corners. The two bases are linked by three distinct lanes referred to as "Top", "Mid", and "Bot." The mid-lane further intersects with the river, dividing the map into four sections of labyrinth-like "Jungle" terrain, which fills the remaining space on the map. [22]

Each team's nexus is positioned near their base and is safeguarded by two turrets. Additionally, each lane on the map contains three turrets and an inhibitor for each team. Players can only attack the enemy nexus if at least one of the enemy inhibitors has been taken down, and inhibitors can only be attacked once all the turrets in their respective lane have been destroyed. While turrets are permanently destroyed once taken down, inhibitors will eventually respawn. In a typical match, one player from each team occupies the top lane, another the mid lane, and two players are assigned to the bot lane. In the bot lane, one player known as the "attack-damage carry" (ADC), serves as the primary damage dealer, while the other plays a "Support" role. The fifth player, called the "Jungler", roams the map, focusing on eliminating neutral minions or neutral monsters found in the Jungle. [65]

The map features several dynamic elements, including minions, neutral monsters, and the champions controlled by the players. Minions are rela-

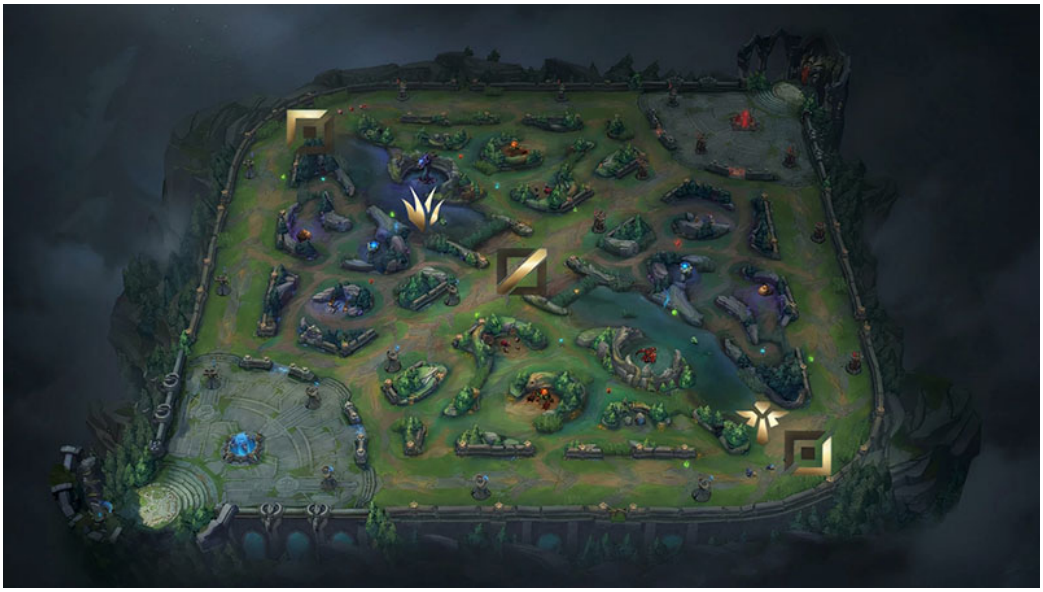


Figure 1: League of Legends Summoner's Rift Map [64]

tively weak non-player characters (NPCs) that, when killed, provide gold to the player who eliminates them [62]. They periodically spawn from each team's nexus and proceed down the lanes in so called "minion waves" or just "waves". Opposing minion waves will always eventually collide at the center of their respective lane if not intercepted by champions or other obstacles. Furthermore, minions will automatically engage in combat with enemy minions, champions, defensive structures, and ultimately the enemy nexus whenever in range. [22]

Each champion in League of Legends is designed to fulfill specific roles, such as Assassin, Fighter, Mage, Marksman, Support, or Tank. Each role brings a unique set of abilities to the game [52]. Assassins are known for their high burst damage and mobility, ideal for swiftly eliminating key targets [103]. Fighters, also known as Bruisers or Off-Tanks, blend damage and durability, excelling in prolonged skirmishes [24]. Mages can cast powerful abilities or spells to inflict damage from a distance [50], while Marksman champions specialize in consistent damage through ranged attacks and abilities [26]. Support champions provide crucial assistance to their team through healing, crowd control, or other utility abilities [69]. Tanks are the frontline defenders, absorbing damage and protecting their teammates [46]. Each role contributes strategically to the team's effectiveness, allowing for diverse and dynamic gameplay.

2.3 Game Objectives

Victory in League of Legends is achieved by eliminating the enemy team's nexus which is guarded by two defensive towers. These two towers, along with the three towers in each lane, must be taken down before the nexus itself can be damaged. Notably, destroying the towers in just one lane is sufficient to make the enemy nexus vulnerable. [22]

Although the primary objective is to destroy the enemy nexus, teams must accomplish various subgoals to pave the way to victory. Apart from the essential task of dismantling towers and gaining advantages by eliminating enemy champions, players focus predominantly on "farming" minions or neutral monsters. This particularly occurs during the so called early-game [24]. Eliminating minions is a critical source of income and experience points (XP). Besides minions, neutral monsters inhabit the jungle, remaining stationary and harmless until attacked [22]. Like minions, neutral monsters offer gold and XP upon defeat. The "creep score" of each player keeps track of how many minions or neutral monsters they have eliminated [24].

The gameplay generally unfolds in three distinct phases: laning, mid-game, and late-game. As mentioned above, players concentrate on farming minions and neutral monsters to accumulate gold and XP during the laning phase [15]. Typically, one player occupies the top lane, another the mid lane, and two players manage the bot lane. Since both teams share these lanes, skirmishes or so called "trades" often occur depending on the players' play styles [22]. The fifth player on each team usually assumes the role of the "Jungler", focusing on farming neutral monsters instead of lane minions. This player occasionally emerges from the Jungle to ambush and kill enemy laners who have strayed too far from their defensive towers [41].

In the mid-game phase, farming becomes comparatively less relevant and players start roaming the map. Here, players leave their respective lanes in order to form groups to eliminate large monsters (so called neutral objectives granting buffs for each team member [69]), destroy towers or pick off isolated enemy champions. It is possible that a player gains an early advantage in gold by repeatedly eliminating enemy champions. This is known as "snowballing" and can cause the game to end in the mid-game phase with a premature victory for the snowballing team [22].

Once the late-game phase is reached, champions are typically equipped with multiple items obtained by spending gold in the item shop [69]. The

item shop is positioned in each team's base and can be accessed by a champion whenever they are located within their respective spawning area. If direct attacks on the nexus have not become feasible at the start of the late-game phase, players might focus on destroying defensive structures or killing "Baron Nashor". "Baron Nashor" (or simply "Baron") is a powerful monster and the most challenging neutral objective in League of Legends. Eliminating it grants substantial stat boosts to the entire team. The game concludes once the nexus is destroyed, after which players enter the postgame lobby to chat and review individual and team statistics. [22]

2.4 Gameplay

A fundamental aspect of gameplay in League of Legends is effective camera control. Players view the game from a top-down perspective and maneuver their champions through a combination of keyboard and mouse inputs. Although controls can be customized, standard practice involves moving the champion, targeting abilities, and controlling the camera with the mouse, while abilities can be activated and items can be used by pressing keys. By the game's default settings, the camera constantly targets the player's champion, however, this is understood to be a disadvantageous practice referred to as "tunnel vision". This means that the player is constrained by merely perceiving their champion's immediate surroundings, limiting their strategic awareness. Skilled players will frequently adjust the camera to monitor their broader surroundings and optimize their positioning accordingly. [22]

Another critical element of gameplay in League of Legends is referred to as "champion mastery" or "mechanical expertise". These terms describe the player's knowledge of their champion's abilities and their synergistic potential [21]. Usually champions have five unique abilities, categorized as either "active" or "passive." Active abilities require manual activation through keystrokes and can trigger actions such as casting spells, creating shields or teleportation. Many activated abilities fall into the category of so called "skill shots". To be effective, skill shots need to be properly timed and aimed by positioning the mouse cursor into a desired direction. Passive abilities on the other hand provide constant stat boosts, like enhanced health regeneration or increased movement speed. [22]

League of Legends also contains some few champions who possess more than five abilities, particularly those that can transform and access different sets of abilities for each form. The overall difficulty of a champion is influenced by the complexity and execution of these abilities, often referred to as

the champion's "kit" [10]. New players may struggle to use abilities efficiently and hit skill shots, whereas experienced players with advanced mechanical skills can fully leverage their champion's kit to defeat opponents. [22]

Since League of Legends provides such a variety of unique champions and abilities, the player community continues to discover new interactions and strategies as different game elements are combined in novel ways [22]. Given these complexities, the argument can be made that League of Legends is a game of construction rather than merely a game of manipulation [5]. Players continually discover and create new strategies, often in ways that even the developers did not anticipate. This ongoing evolution of gameplay elements and player interactions transforms League of Legends into a dynamic and ever-changing environment where innovation and creativity are as important as mechanical skill.

2.5 Ranked Mode

Ranked mode in League of Legends is a competitive gameplay mode, where players are matched based on their skill levels to compete for ranks. Players earn League Points (LP) and Matchmaking Rating (MMR - also known as invisible ELO [48]) by winning matches, which influence their ranking and the skill level of opponents they face. LP is directly tied to a player's visible rank, progressing them through divisions and tiers, while MMR is an underlying rating that determines match fairness and opponent skill level. Winning games increases both LP and MMR and promotes players to higher ranks, while losses result in a decrease, potentially leading to rank demotion. This system aims to ensure balanced and competitive matches which reflect the players' true skill and progress within the so called ranked ladder. [1]

In League of Legends, ranked mode becomes accessible to players once they reach an account level of 30. This milestone is achieved by accumulating account level experience points through gameplay. It is thereby irrelevant if the played games result in victories or defeats. Until players reach this level, they are limited to unranked modes only, which allow them to experiment with and learn the game's mechanics without the pressure of penalties for losses. This restriction aims to fosters a risk-free learning environment for new players. Upon reaching account level 30, players can participate in ranked play using any champion they have unlocked for that account. This includes champions who they might have never played and are thus unfamiliar with. This means that, given that there are 168 champions in the game and it typically takes around 365 games to reach level 30, it is unlikely that

players will develop more than a basic understanding of each champion's mechanics by the time they unlock ranked mode. [22]

Ranked games involve a strategic champion selection process where players take turns choosing their champions. Here each unique champion can be picked by only one player within a game. Before selection begins, each team bans five champions and thereby prevents all players from choosing those champions. This results in a game with 10 banned champions and 10 unique champion picks. This process adds a layer of strategy and requires players to adapt their choices based on the available pool of champions and the opponents' bans. [18]

3 The Meta-Game

In the following subsections, the concept of "meta" in video games is explained, particularly in the context of League of Legends. These explanations include definitions, underlying theories, and the various dimensions of meta knowledge that players must master to gain a competitive edge.

3.1 Definition of Meta

The concept of the "meta" in video games, such as League of Legends, refers to the strategic and tactical knowledge that extends beyond basic game mechanics and that can offer players a competitive edge. Originating from the need to create a framework for complex games involving interdependent global actors [93], the term "meta-game", or just "metagame", is used to describe the broader understanding and contextual strategies employed in video games.

Donaldson proposes a binary model in which the mastering of a game requires two types of expertise: Mechanical expertise and metagame expertise. Mechanical expertise involves skills like interface navigation and character control, and is therefore attached directly to in-game elements. Metagame expertise, on the other hand, involves an understanding of the overarching strategies and the ability to adapt to changes in the game's environment. In the context of League of Legends, this includes staying updated with new strategies, analyzing game updates or patches, and using mathematical techniques to optimize certain champion-kit/item combinations. The argument is made that players who do not utilize this second layer of game knowledge and solely rely on their mechanical skill set, put themselves in a disadvanta-

geous position when facing players with elevated metagame expertise. [22]

Similarly, Rambusch et al. put forward a binary conceptualization in that gameplay can be studied through the physical act of playing and the players' understanding of how the game should be played. On the one hand, there is the physical and motorical handling of the game through electronic peripheral inputs. The second lens, on the other hand, describes how the player makes sense of how the game should be played holistically by considering their role and the game's cultural context. [81]

Tekinbas & Zimmermann's framework for understanding the metagame in video games categorizes its manifestations into four distinct areas: What a player brings to a game, what they take away from it, what occurs in between games, and what happens throughout actual gameplay. What a player brings to a game encompasses the knowledge, skills, and experiences they accumulated from previous gameplay or other relevant contexts. What a player takes away from a game includes the lessons learned and any emotional or cognitive impacts that may affect their future gameplay. What occurs in between games refers to the interactions and activities that occur outside of actual gameplay, such as discussing strategies with other players, analyzing past matches, and staying updated with game patches or community trends. Lastly, what happens throughout gameplay refers to the real-time decisions, adaptations, and interactions that define the immediate gaming experience. [97]

Neuenschwander goes further during investigation of online role-play games by describing the learning aspect of a video game as a multi-faceted proposition. It involves not only the understanding of the formal rules but also the social norms and knowledge specific to the game's community. "They must acquire social capital specific to the subculture, the social norms and skills, social networks, gaming lore and knowledge necessary to function with others in the game's imaginative space." [67][70] This learning process can be facilitated through game books, coaching from other players, imitation, and feedback [70]. Thaicharoen et al. describe meta in video games as a concept revolving around tactical and strategic knowledge that extends beyond the core gameplay mechanics and potentially gives players an advantage over their rivals [99].

Dormans introduces the concepts of ludus (structured play with rules) and paidia (free, explorative play) as opposite extremes of the same spectrum. While certain games are entirely expressed by either paidia or ludus,

most games contain both [23]. In League of Legends, the interplay between ludus and paidia is evident as cultural and gameplay practices transform explorative play into structured strategies and vice versa [22]. Bateman's theory that repeated interactions within the same game environment lead to the development of play patterns and structured strategies [9] also applies here. This type of gameplay knowledge appears to be particularly relevant in multiplayer games, where understanding the sociocultural context is crucial, unlike in single-player games, which typically do not require this form of expertise [22].

Initially, players spend significant time focusing on learning just the game's mechanics [96]. However, as they become proficient in these aspects, they must engage with the game's social context and metagame strategies to progress further. This transition from mechanical to metagame expertise is a crucial milestone in a player's journey, marking their deeper engagement with the game's strategic elements and community [22].

In this thesis, a clear distinction is made between "meta-game" and "game meta" (or simply "meta"), though they are often used interchangeably. "Meta-game" refers to the process of sharing and accessing knowledge that goes beyond basic gameplay, while "game meta" refers to the prevailing strategies players use to optimize their performance in the game.

3.2 Meta of League of Legends

The meta of League of Legends is in a state of constant change, evolving in response to changes within the game and the behaviors of its players [44]. As game updates introduce new elements and balance adjustments, the strategies deemed most effective can shift dramatically [98]. Consequently, what is considered the "best" approach is frequently redefined. Players often describe the current meta with terms like "assassin meta" or "split-push meta", reflecting the prevailing strategies or champion types that dominate the game at a given time [22]. This dynamic nature of the game's meta requires players to continually adapt and refine their tactics to stay competitive.

Since the launch of League of Legends its community and researchers have dissected the game's meta into various subcategories. This breakdown allows players to focus on optimizing and discussing specific aspects of the game. It's important to note that the success of any strategies developed from investigating a specific subcategory of the meta depends heavily on the player's knowledge of other meta subcategories. To be effective, the player

must build a holistic understanding of the game. A selection of the most common League of Legends meta subcategories is introduced throughout the following sub-chapters.

3.2.1 Champion Meta

The concept of champion meta revolves around the performance and effectiveness of individual champions. A champion's strength in League of Legends can vary based on the role they fulfill within their team's composition [37]. Thus, understanding the champion meta also involves recognizing how individual champions fit into and enhance their team's dynamic. A common metric investigated in order to determine the likelihood of winning a game with a certain champion is their win rate. A champion with a win rate higher than 50% is considered advantageous, suggesting it offers players a dominant strategy. Conversely, a champion with a win rate below 50% is seen as disadvantageous. To ensure balance within the game, Riot Games regularly adjusts champions by either nerfing (reducing their power) or buffing (increasing their power) them to keep their win rates around the 50% mark [22]. This ongoing process aims to maintain a fair and competitive environment.

3.2.2 Role Meta

The concept of role meta in League of Legends revolves around the specific roles that champions are designed to fulfill within a team composition. As previously explained, Riot Games identifies six key roles, namely Assassin, Fighter, Mage, Marksman, Support, and Tank [18]. Each of these roles fulfills distinct responsibilities and strengths within a team's composition. The best teams are those whose players effectively cooperate, utilizing their champions to fill the required roles. For instance, this might include a tank to absorb damage or a healer to support allies. Riot Games intentionally designs champions with these roles in mind, however, it has to be noted that a given champion can fulfill different roles depending on how they are played [18]. Together with the champion meta, the role meta heavily influences the positions and lanes to which champions are commonly assigned. Especially in ranked play, it is crucial for players to be aware of this meta aspect to participate effectively as part of their team. Before a game begins, players negotiate position and role assignments which in turn determines where each player will spend most of the laning phase and their responsibilities during the mid and late-game phases. While the framework of choosing appropriate champions for specific lanes is not a formal game rule, it has become part

of the game's etiquette [22]. Players who deviate from these expectations by playing champions in unconventional roles or positions happen to be reported for poor behavior. This reporting behavior stems from the Summoner's Code [77], a set of guidelines created by Riot Games, which emphasizes teamwork and supporting one's team. Many players interpret a refusal to adhere to the meta as a breach of this code [49]. However, Riot Games clarifies that not conforming to the meta is not a violation of the Summoner's Code and should not be considered a reportable offense [95]. This also highlights the ongoing discussion between following established strategies and allowing for creative and unconventional gameplay within the League of Legends community.

3.2.3 Item Meta

The item meta is an aspect of League of Legends gameplay that significantly influences a champion's effectiveness and overall strength. Items in the game are modular upgrades that players can purchase using gold earned during matches [75]. These items can be bought from the in-game item shop and equipped to enhance a champion's abilities. Items purchased in previous games played do not carry over to future games. Items usually grant buffs to champions, enhancing their stats such as attack power, defense, or movement speed, while others offer more unique benefits. The versatility of item choices allows some champions to fulfill different roles based on the items they acquire [18]. For example, a champion typically played as a damage dealer might take on a more supportive or tank role with the appropriate items. This flexibility adds another layer of depth to the game, as players must consider not only their champion's inherent abilities but also how their item choices can adapt their role.

3.2.4 Region Meta

Region meta in League of Legends refers to the unique strategies and playstyles that emerge within different geographic regions. Players are only able to compete against others on the same regional server. The game divides its player base into several regions, including Brazil, Europe East and West, Japan, Latin America North and South, North America, Oceania, Russia, and Turkey [18]. Each region develops its own meta based on the collective preferences and strategies of its players. These regional metas can vary significantly, as different areas might favor particular champions, tactics, or item builds.

3.2.5 Professional Meta

The professional meta (or simply "pro meta") in League of Legends includes the high-level strategies and trends that emerge within the game's global competitive scene. Major competitions include North America's LCS, Europe's LEC, China's LPL, South Korea's LCK, and the PCS, which covers Taiwan, Hong Kong, and Macau, among various other regions. Teams that win in their respective regional leagues earn the opportunity to compete in the annual World Championship, a prestigious event that attracts millions of viewers. For instance, the 2018 World Championship drew 99.6 million unique viewers and featured a prize pool exceeding $6.45 million [42]. Since meta-gaming on the professional level has become increasingly complex and specific, professional teams commonly employ analysts to study their opponents, devise strategies, and develop counter-strategies [42]. Like in traditional sports, top-level players utilize information about their opponents to gain a competitive edge. In League of Legends, preventing an opponent from using their preferred champions is one of the best strategies to undermine them [96]. This is accomplished during the champion selection phase, where players from both teams alternately ban champions, making them unavailable for use by anyone in the match. When the players' identities are known, as in professional play, bans are strategically employed to counter a specific player's preferred play style [96]. This forces them into a role or champion they are less comfortable with. During tournament game broadcasts, analysts, along with shoutcasters, provide detailed reports and insights, enhancing the viewer experience and showcasing the strategic depth of the game [42].

3.2.6 Communication Meta

Communication among team members plays an important role in League of Legends. Players may communicate amongst each other in solo matchmaking using text chat and a ping system, which enables them to mark a vulnerable target or indicate danger on the game map with symbols [22]. Regarding the text chat, players have the option to disable the language filter that is automatically included in the game to prevent swearing and other potentially inappropriate language. Additionally, cross-team conversation ("all-chat") is by default disabled but is configurable, enabling opponent communication throughout the game. Players frequently use psychological tricks and provocations when all-chat is enabled in order to throw off their opponents [22]. Therefore, players have the additional option to silence certain opponents or teammates throughout the game. As opposed to solo matchmaking, premade

teams frequently use external applications such as Skype or Discord for voice communication. This added layer of communication is particularly important for coordinated team play. Because of this, certain champions who are given priority in solo play may not be as attractive in team play, and vice versa. This distinction in importance is exclusive to the MOBA genre [22] and emphasizes how important communication is to gameplay and strategy.

3.2.7 Off-Meta

Off-meta refers to strategies and playstyles that deviate from the current mainstream or meta trends. Although "breaking the meta" can be contentious among players, off-meta approaches can gain acceptance over time, especially if they prove effective [27]. Understanding how the established standard of play operates and how to support or counter off-meta strategies is important in ranked play. Players who master these unconventional tactics can catch opponents off guard, potentially gaining a strategic advantage.

3.3 League of Legends Meta Tools

Meta tools in and around the game of League of Legends enhance players' ability to analyze and understand the game. These online services compile and present data from various categories [102][58][68]. In addition to champion win rates and other statistics, players can for example get information on champion pick frequencies based on region, rank, and game type. Strategic planning and the development of metagame expertise benefit greatly from this abundance of data. Analytical tools such as these are a common around many online multiplayer games across various genres, not just in League of Legends. For instance, aggregate play analytics have shown to be integral to games like Halo Wars, DOTA 2, and World of Warcraft. [22]

Additional to the third-party game statistic platforms, League of Legends also features a "custom" gameplay mode, which allows players to create personalized games. In this mode, players can include friends or computer-controlled bots with adjustable difficulty levels. With no minimum team size requirement, a player can enter a game alone to practice and familiarize themselves with new champions and abilities. Custom mode is also commonly utilized after game updates and patches to investigate the resulting gameplay implications in a sandbox environment. [22]

4 Impact of Game Updates in League of Legends

In the context of video games, "patching" refers to the process of updating an existing game. Although numerous modifications are made before a game's release, post-release patches are integrated into the existing game. Traditional software patching mainly focuses on fixing bugs and enhancing performance of published software [18]. In the video game industry, security patches are frequently used to update code with the aim of preventing players from using known "cheats" or exploits to gain unfair advantages [104]. Today, many game developers use software patches to tweak game balance and introduce new content [18]. In League of Legends these updates regularly modify game rules and balance, thus changing the gameplay experience for existing players [52].

4.1 Game Update Strategy

After more than a decade of development, League of Legends has achieved a high level of maturity in all areas of gameplay. Riot Games provides regular content updates, which helps maintain a large and active player base [57]. Due to the long term attention the game has received by a large player base, one might think that League of Legends could eventually be completely understood and mastered. But this is far from reality. Similar to other online video games, League of Legends is continuously evolving through updates. These patches, being small, mandatory downloads, introduce new content, remove outdated elements, or modify existing code. Typically, players can expect a new patch every two to three weeks, along with a significant annual redesign [22]. This consistent stream of updates ensures that the game remains dynamic and challenging. Riot Games' update strategy for League of Legends aims to balance the elements of exploration and structured play, mentioned above as paidia and ludus. To keep the game engaging and enjoyable, it must allow room for experimentation and creativity. Without this, once players discover the optimal strategies, the gameplay can become monotonous [98]. Regular updates and changes help prevent stagnation by continuously challenging players to adapt and explore new strategies, thereby maintaining the game's dynamic nature [57]. Therefore, Riot Games' patching strategy for League of Legends goes beyond merely fixing bugs or balancing champion strength. The primary goal is to continually alter and refresh the gameplay experience. A Riot Games developer encapsulated this philosophy in one of the published patch notes accompanying each patch, stating:

“Any time we make an update, we’re aiming for two targets: that every game of League feels meaningfully different based on the champions in it, and that each champion in League feels distinct.” [87] And indeed, despite common beliefs, perfectly balanced games have shown to become less enjoyable over time [17]. In this sense, developers do not always have to aim for perfect balance, rather altering the game’s balance in an effort towards maintaining player interest and enjoyment.

To provide an idea of the frequency in which updates to League of Legends are published, from its launch in 2009 up until 2017, Riot Games updated League of Legends more than 160 times, averaging approximately 1.5 patches per month. These updates comprised over 7000 individual changes to the game [18]. These changes include both the introduction of new features and balance changes to the game. These balance adjustments involve tweaking champions’ abilities and occasionally completely reworking older champions [52]. The occasional introduction of new champions and items, as well as the removal of items, can have a significant impact on the overall meta. These infrequent updates tend to create larger shifts in gameplay dynamics compared to regular patches [22]. For reference, League of Legends was initially released with 40 available champions in 2009, and an additional 96 champions were introduced through updates up until 2017 [76].

In League of Legends, patches encompass various types of updates. These include new game content such as newly introduced or revamped champions, items used by champions, and so called skins that alter a champion’s appearance. Balance changes are implemented to adjust the power dynamics among champions, while bug fixes address unintended issues within the game. Additionally, technology improvements are made to enhance performance, both in the game matchmaking client and during gameplay. Overall, patch changes can be categorized into three main types: gameplay changes, bug fixes, and visual updates. Gameplay changes impact how champions interact with the map and perform against opponents, adjusting their abilities and overall balance. Bug fixes correct errors in the game, such as for example issues that prevent champions from casting spells if an action is interrupted. Visual updates enhance the game’s aesthetic, including modifications to the map, champion graphics, skins, and other visual effects. [18]

4.2 Public Beta Environment

Riot Games utilizes a "public beta environment" (PBE) to test most of the changes included in each League of Legends patch. This test server allows

players and developers to experiment with upcoming updates before they are officially released. Players must apply to join the PBE, and testers are rotated every few weeks to ensure unbiased perspectives. Participants can offer feedback on new changes and report bugs through the PBE forum, which is accessible to the public. The feedback gathered from these testers often leads to further adjustments or reversals of changes before they are officially implemented as game patches or updates on the main servers. This iterative process helps refine updates and ensures a smoother balancing experience for all players when the patches go live. [22]

4.3 Gameplay Impact

League of Legends features a wide variety of unique champions and abilities, leading to unpredictable interactions when new elements are added or existing ones are changed. The game's complex mechanics allow players to continuously discover new synergies and conflicts by experimenting with different combinations. This exploration keeps the gameplay engaging and ensures that updates can have significant impacts, often revealing unexpected results that require further adjustments by the developers to maintain balance and competitive integrity. [22]

Patches in League of Legends have significantly expanded the game's content, more than doubling the number of champions available since its release and frequently adjusting the game's balance for active players [18]. Analysis by Claypool et al. [18] reveals an unusually high rate of updates compared with what is considered to be mature software, focusing primarily on gameplay adjustments rather than fixes or visual enhancements. These updates aim to balance the game by effectively boosting the win rates of underperforming champions and reducing those of overperforming ones. New content like champions, items, abilities, and maps forces players to relearn mastered aspects of the game. Although balance changes are easier to adapt to since the core gameplay remains unchanged, they still require strategic adjustments. Gameplay changes are crucial for addressing balance issues and can be categorized into numerical changes (altering stats), utility changes (modifying ability interactions), and quality of life changes (enhancing champion usability). Each gameplay change can be a buff (increasing champion/item strength), a nerf (decreasing champion/item strength), or neutral (neither clearly strengthening nor weakening the champion/item). For example, a buff might increase a champion's base armor, a nerf might reduce the radius of a spell, and neutral changes could adjust one aspect while balancing another. Significant additions from patches include new champions, typically

added one at a time, with the number of releases varying by season. The win rate, reflecting a champion's effectiveness, guides adjustments throughout the following updates. Champions with win rates significantly above or below 50% are frequently adjusted to maintain balance. Riot Games strives to keep win rates around 50% through buffs or nerfs, ensuring no champion consistently dominates or is at a disadvantage. [18]

4.4 Economic Impact

Throughout the upcoming subsections, the business model of League of Legends is introduced. The focus is laid on its digital product structure and the freemium approach, followed by a discussion on how continuous updates help maintain player engagement and drive revenue through virtual goods.

4.4.1 Business Model of League of Legends

League of Legends operates as an information product, leveraging the unique cost structure of digital goods. Unlike physical products, the primary expenses for League of Legends are concentrated in development, which includes design, programming, and marketing. Although these upfront costs are substantial, once the game is created, the marginal cost of reproducing and distributing the software to additional users is almost negligible [89]. This dynamic of digital goods enables significant economies of scale, where the per-unit cost of the software decreases as more users join. As the game's infrastructure improves over time, maintenance and development costs can be further reduced. [7]

League of Legends follows the freemium model, which combines free access to the core service with monetization through optional premium versions of the software [51]. In the context of video games, freemium and free-to-play are often used interchangeably [2]. The free-to-play model allows players to download and play the game at no cost, attracting a large player base due to its accessibility. The primary difference between freemium and free-to-play lies in microtransactions, where players can buy virtual items such as skins, champions, and other cosmetics [16]. Because the game itself is free, players are more likely to spend money on these virtual items [105], often through a per-item billing model. This model has proven successful in growing a large user base and creating a strong community, even if it means temporarily sacrificing profitability [72]. The network effect, where players recommend the game to their friends, occurs likewise for players who do not intend to purchase anything [92]. A small percentage of players drive revenue through

in-game purchases of virtual goods [63], which has proven to be the most effective way for developers to generate revenue within this free-to-play model [7].

In League of Legends, virtual goods encompass objects like skins, champions, in-game currencies and tokens. These virtual goods are essentially simulations of real-world objects within the game's virtual environment [54]. Generally, they fall into two main categories: functional and decorative [61]. Functional virtual goods, such as items or characters, typically offer gameplay advantages, while decorative goods, like skins and emotes, are purely aesthetic and do not affect gameplay mechanics [55]. The pay-to-win model, where players can purchase functional items to gain an edge over others, is heavily criticized because it undermines competitive integrity and creates an unfair playing field [2]. To maintain fair competition, purchasable virtual items in League of Legends are designed to be non-essential and do not directly impact gameplay [52][61], ensuring that all players have an equal opportunity to succeed based on skill rather than financial investment.

A percentage of players spend money on cosmetics in League of Legends due to the influence of social norms and self-presentation. Players often value how they are perceived by others in the game, using cosmetic items to express their desired self-image and influence others' perceptions [45]. This identification with the game and its community makes demonstrating one's personality and status through purchases meaningful. The desire for online self-presentation, where players aim to project a preferred image within the virtual community, is a key factor driving the purchase of digital items [43]. Additionally, some players purchase cosmetics out of an altruistic motive, with an aim of supporting the developers for providing a free game and contributing to its continued development and improvement [35].

4.4.2 Revenue through Updates

Continuous updates are published in League of Legends to maintain and enhance its attractiveness to players. These updates include both changes to the gameplay aspects (thereby shifting the meta), but also new cosmetic items which can be purchased as virtual goods. While introducing new purchasable virtual items can diminish the value of existing ones, these updates keep the game dynamic and engaging [36]. Riot Games frequently releases updates based on user feedback [38], primarily in the form of patches. Each patch is accompanied by detailed patch notes [31] that outline all game changes. This consistent flow of updates is important for the game's long-term success,

ensuring that it remains interesting, balanced, and responsive to the player base's needs [36].

Ling and Wang state that the gaming industry is highly competitive and to ensure that players don't drift towards other games, developers must consistently deliver perceived value. This value is seen in the quality of the product, the service provided, the system's reliability, and the pricing model [56]. Regular updates additionally give players a reason to log in frequently to see what has been added to the game [36]. This anticipation and curiosity drive players to stay connected with the game, ensuring that it remains a part of their regular entertainment routine. Multiple studies have shown that spending more time in the game leads to a higher likelihood of players spending money on virtual goods [40][33][19]. When players are consistently engaged, they become more invested in the game, both emotionally and financially [7]. In-game purchases, such as skins, champions, and other cosmetic items, are therefore more appealing to players who feel deeply connected to the game and its community. Hsiao and Chen investigate player loyalty as another essential element influenced by continuous updates. Players who feel that the developers are committed to improving the game and listening to their feedback are more likely to develop a sense of loyalty towards the game. Ultimately, a loyal player base is more inclined to support the game financially, as they feel a personal connection and a sense of belonging within the game's community [39].

5 Reddit and League of Legends

In the following subsections, the role of Reddit as a platform for community-driven discussions and knowledge exchange is showcased, particularly within the context of gaming. First, the structure and features of Reddit are described before highlighting its significance in fostering meta discussions in the League of Legends community.

5.1 Reddit Platform

Reddit is a social network and news platform where users can share content, engage in discussions, and stay updated on a wide range of topics [25]. Unlike traditional social networks where users follow other users, Reddit operates around communities called "subreddits" [66]. These are essentially forums centered around specific topics, interests, or themes, ranging from hobbies and news to more niche subjects like literature or specific video games. Each

subreddit is a self-contained community with its own set of rules, created and enforced by the users who moderate that particular subreddit, while the platform itself is governed by site-wide policies set by Reddit [25]. One of the key differences between Reddit and other social media platforms is the way users interact. On Reddit, you users do not follow individual users, but rather the subreddits that interest them [66]. This means a users feed is filled with content related to the topics they care about, rather than updates from specific people. This structure fosters more topic-focused discussions and can create a sense of community among users with similar interests. Reddit has become one of the most popular social networks in the world [79], hosting over 100,000 active communities [82] and more than a million subreddits [25] in total. Since October 2023, Reddit has over 70 million daily active unique users [82], making it a significant platform for user-generated content. With its tagline "the front page of the internet", Reddit aims to be a central hub where new and popular content can be shared on the internet [4].

Reddit is a platform where not only the content, from posts to comments, is user-generated but also the popularity of this content is determined by the users [4]. When users come across a post or comment, they have the option to upvote or downvote it, which affects its visibility on the site [66]. The more upvotes a piece of content receives, the more likely it is to appear at the top of a subreddit or on the front page of Reddit. Anyone can register for a Reddit account for free, however, Reddit also offers a premium membership called "Reddit Gold" [4]. For a monthly or yearly fee, Reddit Gold provides users with additional features like for example ad-free browsing. Users don not have to be a registered to read or discover content, but they do have to register in order to post and vote [4]. Registered users can share two main types of content: text posts, which are known as "self posts", and links [66]. These links can include a wide variety of media, including videos, pictures and articles [4]. The content on Reddit is ranked by the community through upvotes and downvotes [66]. Posts with more upvotes move up in visibility, while those with more downvotes can fade from view. The same voting system applies to comments [4], which are a central part of Reddit's discussions. Comments form what's known as a discussion tree: the original post acts as the root, and each comment is a branch that can have its own sub-branches if others reply to it [66]. When users post content that is upvoted by others, they can earn "Karma Points" [4]. This score reflects how much the community appreciates a user's contributions. If content violates subreddit rules or Reddit's site-wide guidelines, users can report it [25]. Moderators, who are users themselves, have the authority to remove posts and even ban users who consistently break the rules [25].

5.2 Knowledge Exchange in the Gaming Community

Multiple studies have confirmed the integral relationship between video game play and the broader sociocultural contexts in which it occurs [22]. It can be argued that gaming does not happen in isolation but rather within a community where players draw upon their previous experiences and external resources to succeed [60]. This is particularly evident in games like League of Legends, where participation in the game is deeply intertwined with community involvement. When players engage with a game, they don't just interact with the game's software; they also participate in a larger cultural phenomenon. This includes sharing tips, strategies, and user-created content, which help build and sustain a community. For example, after the release of Doom in 2008, players began exchanging custom levels in online forums. Such activities are not merely supportive but are crucial in fostering a sense of belonging among players, leading to the creation of what is known as a participatory culture [5]. In this culture, players not only consume game content but also produce artifacts such as mods, guides, or strategies that contribute to the community's knowledge and evolution [78].

The idea of metagaming, where players develop strategies beyond the basic game mechanics, is particularly relevant in League of Legends due to its complexity and the vast number of unique champions and abilities [22]. As players experiment with different combinations and interactions within the game, they often discover strategies and mechanics that were not initially apparent or even intended by the developers [5]. In League of Legends, metagame expertise becomes crucial at the highest levels of play, particularly in ranked modes, where understanding the meta can significantly influence the outcome of matches [22]. As players reach the minimum level to enter ranked mode, they are expected to have not only mastered the game's basic mechanics but also acquired a deep understanding of the current metagame. This knowledge is necessary to make informed decisions during champion selection and in-game strategy, which are critical to success in competitive play. The continuous learning process that players undergo, even at the highest levels, reflects the dynamic nature of the game and its community. As the game evolves, so does the meta [44], driven by the collective efforts of the player base and developers. This ongoing exchange of game-related knowledge within the community underscores the importance of sociocultural factors in gaming, as players collaboratively shape and redefine the game over time [5]. Understanding this process is essential not only for analyzing current

game cultures but also for designing games that encourage and facilitate such extrinsic play.

5.3 League of Legends Meta on Reddit

The meta exchange of the League of Legends player base on Reddit revolves around a collaborative and analytical approach to improving gameplay. Players engage in discussions on subreddits like "LeagueofLegendsMeta" or "SummonerSchool", where they dissect and debate the effectiveness of specific items, strategies, and champions. These discussions involve both theory crafting, where players hypothesize how different game elements interact, and detailed analysis, which includes in-depth calculations and strategic breakdowns of recorded gameplay moments. Reddit, along with other resources like written guides and video tutorials, becomes an important platform for players to enhance their understanding of the game without waiting for in-game experiences to naturally occur [22]. It allows players to preemptively prepare for various situations by learning from others' experiences and analyses. The popularity and activity within these subreddits highlight that metagaming in League of Legends is not just an individual endeavor but a collaborative one [60]. Players contribute to and benefit from a collective pool of knowledge, which accelerates the learning process for both active contributors and passive readers. This creates a knowledge hierarchy within the online community, where some players are recognized for their expertise and their contributions are given more weight [47]. Additionally, discussions on Reddit frequently explore off-meta strategies that deviate from the conventional meta. These off-meta strategies can sometimes lead to success in gameplay and their viability is frequently debated in these forums. As a result, Reddit serves as a platform where both standard and unconventional strategies are scrutinized, discussed, and refined by the community, making it an essential part of the broader League of Legends metagaming experience. [53]

6 Methodology

With an aim to answer the research question, the following chapters illustrate a data scientific approach coupled with domain-knowledge-based interpretation. It is important to understand that extracted meanings from anonymous user generated content are by nature volatile in veracity due to different cultural contexts, different levels of understanding about the game of League of Legends, and emotionality. It is therefore necessary that any information,

that is extracted and aggregated as a result of the following methodology, is interpreted by an actor with extended access to League of Legends domain knowledge. In essence, the following approach aims towards proposing a way of informing future game updates based on past user expressions and feedback on the Reddit platform.

6.1 Overview

A simplified overview diagram of the research methodology can be seen in Figure 2. A more detailed overview diagram can be found in Appendix A. In order to gather user generated content about League of Legends from Reddit a list of appropriate subreddits is defined, as outlined in Section 6.2.1 hereafter. Then a content scraper is developed (Section 6.2.2) which accesses the Reddit API and requests content and metadata of posts published to the subreddits. After subjecting the collected posts data to an initial data analysis (Section 6.2.3), the dataset is trimmed to contain only posts from time periods when League of Legend patches 14.1 to 14.8 where active. As described in section 6.2.5, its comment trees for each remaining post are extracted and comment contents and metadata are additionally queried through the Reddit API. Additionally, Reddit post contents are analyzed using TF-IDF clustering techniques (Section 6.3) in order to identify prominent concepts that are either unique to each patch (Section 6.3.3) or common (Section 6.3.4), meaning they are discussed throughout all patches. Furthermore, in order to specifically target League of Legends items and champions in the meta analysis an item dictionary and a champion dictionary is manually set up, as described throughout Section 6.4. Two different sentiment classification models are applied to the contents of the Reddit posts and comments in order to efficiently evaluate sentiment scores (Section 6.5). Together with the established dictionaries and TF-IDF terms, the resulting datasets containing Reddit posts and comments and corresponding sentiment scores form the data foundation for the following insights analysis and results interpretation in Section 7. Additional aggregated champion metadata (particularly win rates, pick rates, and ban rates) are extracted for patches 14.1 to 14.8 from three selected websites (Section 6.6). Section 6.7 describes how Riot Games' Data Dragon service is utilized to collect various champion statistics for each champion for the selected patch durations. These champion statistics and aggregated champion metadata can be used to effectively compare and interpret changes in League of Legends meta discussions on Reddit in Section 7. Various code artefacts and datasets established throughout this methodology can optionally be accessed via the referenced GitHub repository [34].

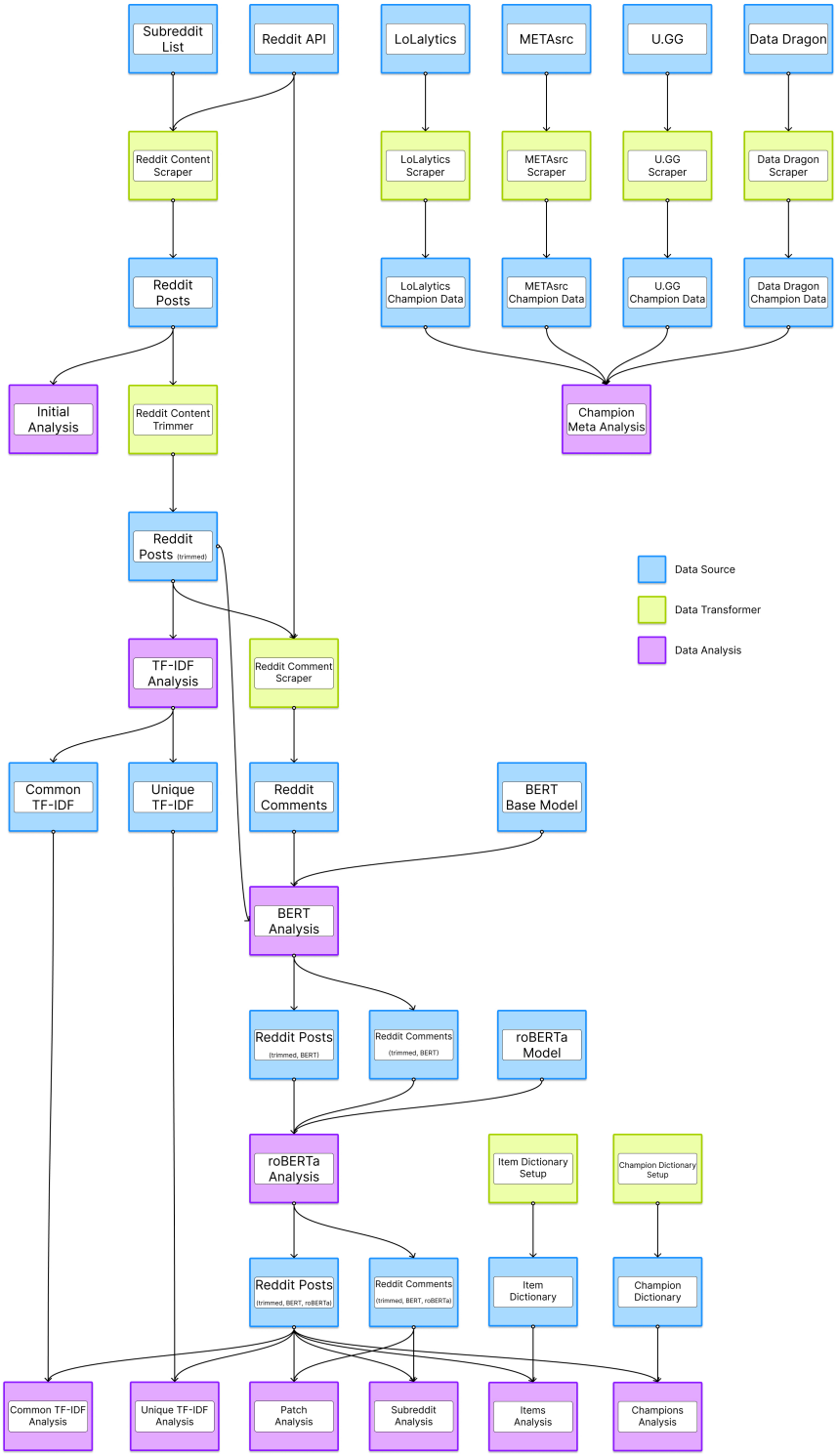


Figure 2: Research Methodology Overview

6.2 Reddit Datasets

The following sub-sections describe the process of defining relevant subreddits and developing a content scraper to collect posts and metadata using the Reddit API. An initial data analysis is applied to the collected data which is then filtered to focus on posts related to specific League of Legends patches. Sub-sections also cover the extraction of comment trees and additional querying of comments from the Reddit API.

6.2.1 Subreddit Selection

When conducting data collection of Reddit posts about League of Legends, it is necessary to define a specific list of subreddits beforehand to ensure the relevance and accuracy of the data collected. Reddit hosts a large number of subreddits, each with its unique focus and user base. The predefined list aims to prevent the data collection process to include irrelevant or tangential discussions, diluting the quality of the analysis. By selecting specific subreddits, conversations can be targeted that are directly related to League of Legends. The complete list of 166 subreddits used to extract content in the scope of this research approach can be inspected through Appendix B.

League of Legends subreddits can be categorized based on the type of meta discussed within each subreddit. For instance, general meta subreddits like "summonerschool" cover broad topics and strategies that are applicable across various aspects of the game. Champion class meta subreddits, such as "MarksmanMains", focus on strategies and discussions related to specific classes of champions, like Marksmen. Role meta subreddits, including "supportlol", center around gameplay and strategies for specific roles within the game, such as the Support role. Additionally, there are subreddits dedicated to the meta of individual champions, such as for example "AatroxMains", which provides discussions and strategies tailored to a single champion. The "LeaguePBE" subreddit is dedicated to the Public Beta Environment version of League of Legends, where users discuss potential future changes to the game before they are implemented in the main version.

6.2.2 Reddit Posts Data Collection

A Reddit content scraper is developed to collect and update post contents and metadata from the various League of Legends related subreddits outlined in Appendix B, using the Reddit API through the 'praw' library [12]. The scraping of the posts is conducted during the months of April and May of 2024 and includes the collection of posts that were created even before

the beginning of that year. The scraper includes a function that converts and stores timestamps of when each post and metadata is collected. After a connection to the Reddit API using the provided credentials is established and an empty Pandas DataFrame [73] is initialized.

The scraper iterates over the predefined list of subreddits, retrieves the most recent posts from each subreddit, and stores relevant information (like title, author, score, comment count, etc.) in the DataFrame. It also updates the score and comment count for existing posts and adds new posts that were not previously in the dataset. Finally, the scraper performs consistency checks to ensure the uniqueness of post IDs and to count missing values, before saving the updated dataset back to a CSV file. For each post, its subreddit, title, ID, full ID, author, URL, upvote score, comment count, and content is stored along with three timestamps for when the post has been created, collected and updated as well as a binary classification variable marking self-posts. In total, data from 180607 individual posts is collected.

The Reddit API operates by allowing users to query posts from a subreddit in a manner that effectively moves "backwards in time" along the subreddit's timeline. This means that when posts are requested, the API returns the most recent posts first, and as querying is continued, it provides older posts. However, there is a quantitative limit on how many posts can be retrieved across multiple queries from a given subreddit. Because of this limit, the oldest (or furthest back) timepoint accessible varies between subreddits. This discrepancy creates a challenge when aiming to analyze data from multiple subreddits over a consistent time period. In later stages of data processing (detailed in Section 6.2.4), this issue is addressed by trimming the collected content from all subreddits to a common time period, ensuring that the analysis is conducted on a temporally consistent dataset across all subreddits.

6.2.3 Initial Data Analysis

The next step is to inspect the dataset containing the collected Reddit posts and select a common timeframe. This is accomplished by loading the dataset of Reddit posts and categorizing the posts according to specific League of Legends patch versions. First several time periods corresponding to different patches of the game are defined, spanning from Patch 14.1 to Patch 14.9 (dependant on the point in time when the data collection process was conducted). These time periods are used to assign a patch number to each Reddit post, based on the creation timestamp. Posts created outside the

defined patch periods are assigned a value of 0, indicating they don't belong to any of the specified patches. The total number of posts collected for each subreddit, regardless of patch versions, can be seen in Appendix C.

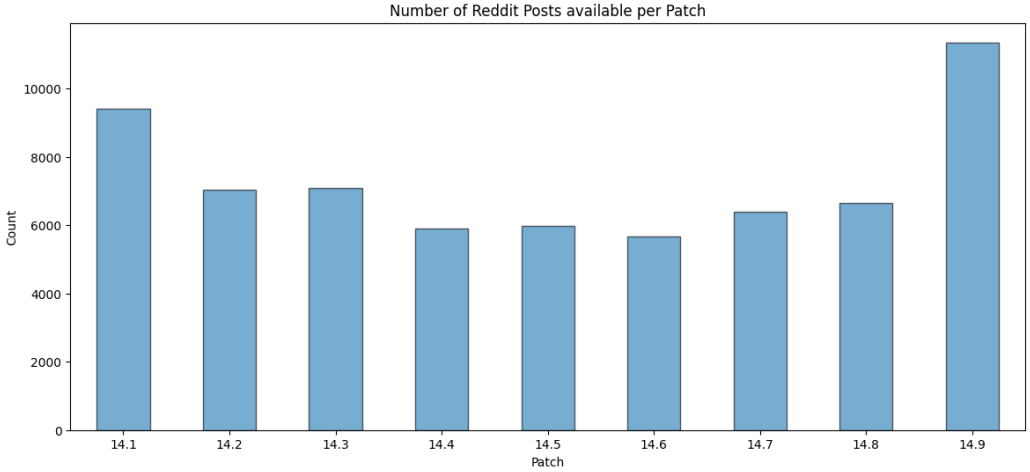


Figure 3: Number of collected Reddit Posts per Patch

Figure 3 displays how many posts are associated with each patch. This provides an overview of the distribution of Reddit discussions across different game patches, helping to identify which patches generated more community engagement on Reddit.

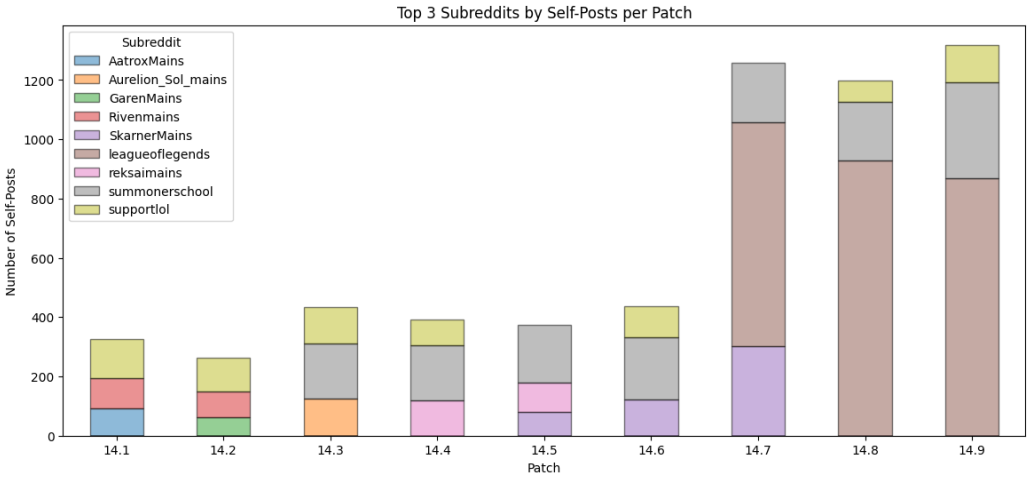


Figure 4: Top 3 Subreddits per Patch ordered by the Number of collected Reddit Self-Posts per Patch

Since self-posts are particularly interesting, due to their machine readable textual content, the dataset can be filtered to include only those posts that are classified as such. Figure 4 displays the top three subreddits for each patch in terms of the number of self-posts. This is achieved by grouping the data by patch and then selecting the top three subreddits with the highest counts within each patch. The result lists the most active subreddits (based on self-posts) for each patch.

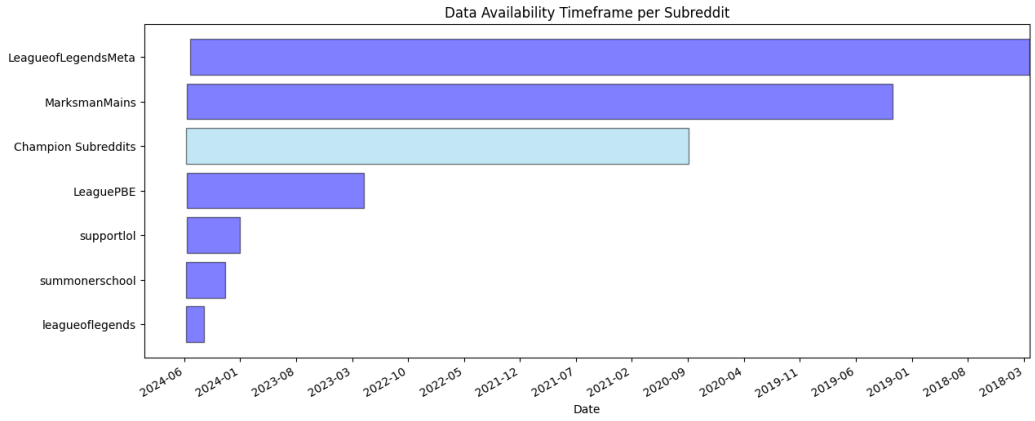


Figure 5: Data Availability Timeframes for each Subreddit

In order to visualize the data availability timeframe for each subreddit within the dataset all Reddit posts are grouped by subreddit and both the oldest and newest creation timestamps within each subreddit are evaluated. These timestamps represent the earliest and latest posts available for each subreddit in the dataset. The duration of data availability is then calculated by subtracting the earliest post date from the latest post date. Figure 5 visualizes the duration and dates of data availability for each subreddit. Note that for compactness reasons, all subreddits centered around a specific League of Legends champion have been summarized and are visualized in the chart via the light-blue bar labelled "Champion Subreddits". Within this subset, the earliest timestamp corresponds to the earliest timestamp of all champion specific subreddits, and analogously, the latest timestamp corresponds to the latest timestamp of all champion specific subreddits.

To assess data availability across different League of Legends subreddits, focusing on specific patch periods each patch version is mapped to its release date (Appendix D). These timestamps therefore represent the start of each patch period. The visualization in Figure 6 is zoomed in on a specific time period, set from January 1, 2024, to May 1, 2024, providing a focused view

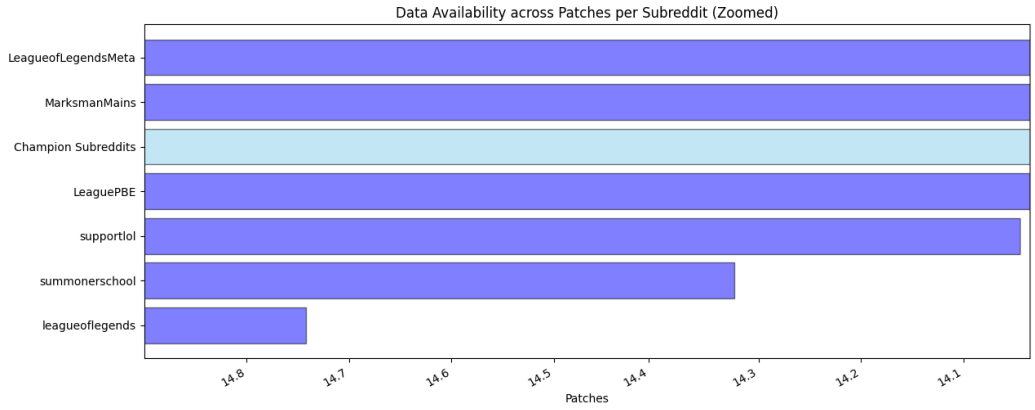


Figure 6: Data Availability Timeframes for each Subreddit

of data availability across the defined patches for each subreddit.

6.2.4 Posts Dataset

Since the data collection process takes place during the duration of the 14.9 patch, patches 14.1 to 14.8 are selected for further analysis. They represent the first eight patches of the 14th season of League of Legends, spanning from January 10th to April 30th, 2024. This time frame allows for a comprehensive examination of player discussions and trends across the initial phase of the season, capturing how the community responds to and adapts to new changes introduced by these patches. Analyzing this specific period can provide insights into the evolving meta and the community's engagement with significant updates early in the season. The temporally trimmed posts dataset contains 54129 posts in total out of which 31075 are self-posts.

6.2.5 Comments Dataset

In order to collect and store comments from the Reddit posts in the posts dataset, a Reddit comment scraper is developed similar to the content scraper. It specifically targets posts that have at least one comment. First a DataFrame with predefined columns is set up to store information about each comment, such as the parent post ID, comment ID, author, upvote score, and content along with two timestamps for when the post has been created, collected. The scraper loads the CSV file containing the dataset of Reddit posts which has been filtered and trimmed to focus on the specific time period. The scraper then iterates through the list of post IDs in the dataset that have a non-zero comment count and each post its comments are retrieved using

the Reddit API via the "praw" library. Comments are only stored in the comments dataset if they contain all the necessary attributes (ID, author, score, creation time, and content). This is done to avoid the collection of comments that have been deleted or comments that where posted by accounts that have been deleted at the point of collection. To adhere to Reddit API rate limits, the scraper interrupts the collection process for 0.6 seconds between processing each post. This means that the comment scraper has to run for an extended time in order to collect all comments from the selected posts. Therefore a checkpoint is saved to a CSV file after processing every 100 posts, ensuring that progress is not lost if the scraper is interrupted. In total, all 329088 comments are collected with the scraper running for 3,88 days.

6.3 TF-IDF Clustering

TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical technique used to identify and extract prominent words from a collection of text documents, such as Reddit posts. It works by evaluating the importance of a word within a specific document relative to its frequency across all documents in the dataset. The term frequency (TF) measures how often a word appears in a document, while the inverse document frequency (IDF) decreases the weight of common words that appear across many documents, thereby highlighting words that are more unique or significant to specific documents.

TF-IDF clustering is a technique that combines the power of TF-IDF with clustering algorithms to group similar documents based on the prominent words they contain. After computing the TF-IDF scores for each word in the dataset of Reddit posts, each post is represented as a vector in a high-dimensional space, where each dimension corresponds to a word with its associated TF-IDF score. These vectors capture the essence of each post by emphasizing the words that are both frequent within the post and distinctive across the dataset. Clustering algorithms, such as K-means, are then applied to these TF-IDF vectors to group posts into clusters based on their similarity. Posts within the same cluster share similar patterns of prominent words, indicating that they discuss related topics or themes. TF-IDF clustering is particularly useful in this context because it allows for the automatic discovery of underlying themes or topics in the large collection of documents, helping to categorize and analyze the content of Reddit posts in a more structured and meaningful way.

While in traditional TF-IDF clustering, the focus is on clustering docu-

ments by treating each document as a vector of term weights, in this research approach the perspective is switched to treat each term as a vector of document weights (where the vector components represent how important the term is across different documents). Then the terms are clustered based on how similarly they distribute across the document set. This method reveals groups of words that tend to appear together or have similar importance patterns across the dataset, indicating they are related in context. This is useful for identifying key themes, synonyms, or related terms within the corpus or Reddit posts. Essentially, with this approach, the vocabulary of the dataset is clustered rather than the documents, which results in clusters of distinct words instead of clusters of documents.

Using TF-IDF term clustering is particularly effective in the context of analyzing Reddit posts because it allows us to identify words and concepts that are not only frequent but also distinctive within the context of certain discussions. This helps in surfacing key terms that are highly relevant to particular posts or subreddits while filtering out common words that are less informative. As a result, TF-IDF term clustering provides a more nuanced understanding of the textual data, making it ideal for uncovering trends, themes, and topics within the large and diverse set of Reddit posts being analyzed.

6.3.1 Term Cluster Separation

When performing TF-IDF term clustering, terms happen to overlap between clusters, thereby forming duplicates. Depending on the number of clusters, which is manually defined prior to clustering, the percentual overlap varies. Therefore the overlap of TF-IDF terms across different clusters is assessed as the number of clusters (k) is varied, in order to find the optimal number of clusters prior to performing the cluster analysis. Firstly, functions are defined to clean the text of the Reddit posts, removing stopwords, punctuation, and digits, ensuring that only meaningful words are analyzed. Then a TF-IDF vectorizer is implemented to convert the cleaned text data into numerical representations (vectors) that reflect the importance of each word in relation to the others across all posts. These vectors are then used as input for the K-means clustering algorithm, which groups similar terms together based on their TF-IDF values. After clustering the terms into a specified number of clusters, the code extracts the top terms from each cluster and stores them in a DataFrame. This process is repeated for different values of k, ranging from 3 up to 20, to analyze how the number of clusters affects the overlap of terms across clusters. Specifically, the number of duplicate terms (terms

that appear in multiple clusters) and their percentage of the total terms is calculated with each iteration.

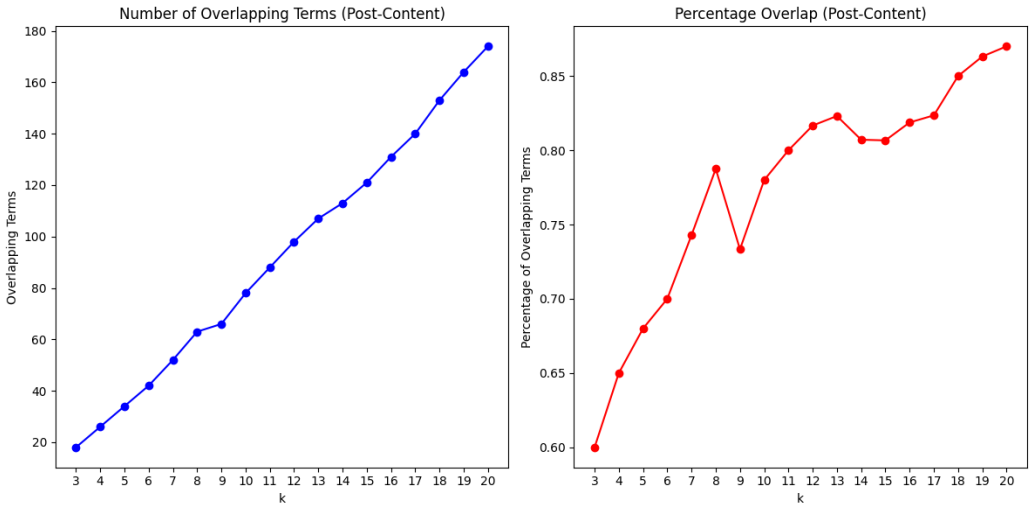


Figure 7: Number of overlapping Terms and Percentage Overlap across Post Content Term Clusters for varying k

The results are visualized in Figure 7 showing the absolute number of overlapping terms and the percentage of overlapping terms as k changes. These visualizations help to understand the impact of varying k on the distinctiveness of the clusters, providing insights into the optimal number of clusters for minimizing term overlap and enhancing the clarity of the clusters. In this context, an elbow-method-like approach is applied to determine the optimal number of clusters by analyzing the percentage overlap of terms across different cluster sizes. As the number of clusters is varied, the percentage overlap line typically exhibits a pattern where it first begins to increase as k continues to grow, reflecting that terms start overlapping more across clusters. The optimal number of clusters is then identified by looking for a low point in the percentage overlap line after it has initially increased. This low point (in this case at k=9) suggests that at this particular value of k, the clusters are relatively distinct from one another, while still allowing for a sufficient number of topical clusters. The exact numbers of duplicates, duplicate percentage and top 5 duplicate terms for each k can be observed in the table contained in Appendix E.

6.3.2 Patch-Specific Terms

Firstly, the goal is to capture and compare prominent terms across different patches of the game. A DataFrame is initialized with columns corresponding to the patch number and the clusters. Then iteration over each unique patch in the dataset is started. For each patch, all Reddit posts associated with that patch are selected, the text of those posts is cleaned, and TF-IDF clustering is performed with an optimal number of clusters. The top 100 terms from each cluster are extracted and stored. For each patch, these clusters of terms are compiled into a single row, where each column corresponds to one cluster, and the terms are joined into a comma-separated string. In a next step the term clusters for each patch are condensed into a single set of unique terms. This final DataFrame provides a summary of the distinct (i.e. distinct within patches and not across patches) terms associated with each patch, helping to identify and compare key terms across different game updates.

6.3.3 Unique Terms

In an effort to identify and extract unique terms associated with each patch in the dataset a new DataFrame is created with columns for the patch number and the unique terms associated with that patch. Then all terms from the original condensed cluster terms column are collapsed into a single list. A Counter object is used to count the occurrences of each term. Terms that appear more than once across different patches are considered duplicates. Next, the code iterates through each row of the original DataFrame, which contains the terms for each patch. For each patch, it filters out the duplicate terms identified earlier, leaving only the unique terms specific to that patch. These unique terms are then combined into a comma-separated string and mapped with the corresponding patch number. The resulting DataFrame (Table 1), contains only those terms that are distinct to each patch, providing a clearer view of the specific vocabulary or themes that emerged uniquely in each game update.

Patch	Unique Terms
14.1	hydra, crit, lethality, far, great, armor, gives
14.2	buff
14.3	ve, thanks, tips, change
14.4	patch
14.5	looking, eq, eqeq, tell
14.6	shield
14.7	wave, slow, old
14.8	bot, wanted, instead

Table 1: Unique TF-IDF Terms per Patch

6.3.4 Common Terms

Finally, terms that are common across all patches in the dataset are identified. Terms associated with the first patch in the original cluster DataFrame are stored in a set. This set initially contains all the terms from that first patch. Then the script iterates over the terms associated with each subsequent patch. For each patch, the terms are split into a list and an intersection update method is used to modify the set by retaining only those terms that are present in both the current set and the terms from the current patch. This process effectively narrows down the set to include only those terms that appear in every single patch.

These terms are 'want', 'actually', 'guys', 'kill', 'hit', 'mid', 'game', 'got', 'really', 'played', 'ult', 'champion', 'xb', 'support', 'like', 'main', 'feel', 'use', 'make', 'going', 'love', 'hp', 'help', 'early', 'play', 'dont', 'item', 'know', 'enemy', 'maybe', 'feels', 'team', 'playing', 'adc', 'ive', 'speed', 'level', 'games', 'bad', 'jungle', 'way', 'fun', 'items', 'people', 'better', 'ad', 'getting', 'pretty', 'used', 'new', 'think', 'try', 'season', 'build', 'damage', 'lane', 'time', 'win', 'gold', 'need', 'lot', 'champ', 'hard', 'champs', 'ap', 'usually', 'good', 'best', 'right', and 'passive'.

6.4 League of Legends Dictionaries

Going beyond the terms and concepts identified throughout TF-IDF analysis and in order to more specifically target League of Legends item and champion meta, dedicated dictionaries are manually set up. These allow for the precise identification and extraction of specific in-game items and champion names, that may not be fully captured or highlighted during the TF-IDF analysis. While TF-IDF is effective for identifying prominent terms

and themes across Reddit posts, it may overlook or underemphasize specific items or champions due to their contextual usage or the presence of similar terms. By incorporating these dictionaries, further relevant in-game elements are systematically identified and analyzed, providing a more comprehensive update-oriented understanding of discussions and trends related to specific items and champions in the game.

6.4.1 Item Dictionary

An item dictionary is set up to map all synonyms and aliases of League of Legends items to their corresponding single item name. This dictionary (Appendix F) covers all 199 items available in the regular 5v5 game mode, providing comprehensive coverage of the game's item system. In total, the dictionary contains 916 aliases as values, capturing the various ways players may refer to items, including abbreviations, misspellings, and colloquial terms. This mapping allows for consistent and accurate identification of items across discussions, aiming to enhance the precision of subsequent analysis.

6.4.2 Champion Dictionary

Furthermore, a champion dictionary is established, mapping all synonyms and aliases of League of Legends champions to their respective single champion names. This dictionary (Appendix G) includes all 166 champions available in the regular 5v5 game mode, making sure that every champion is accurately represented. Similar to the item dictionary, the champion dictionary captures the various ways players refer to champions with a total of 935 aliases as values. This allows for consistent identification of champions across discussions.

6.5 Sentiment Classification

All posts and comments have been processed through two selected sentiment classification models to compute sentiment scores, providing data for further analysis of community sentiment across different discussions and patches. Computing sentiment scores with two different sentiment classification models is useful because it allows for cross-validation and comparison of results, providing a more robust understanding of sentiment by leveraging the strengths and characteristics of each model.

6.5.1 Sentiment Scores as Proxies

With this approach the assumption is made that sentiment scores of League of Legends Reddit posts and comments represent proxies for gauging player satisfaction or dissatisfaction with the current meta, but this comes with important caveats. Sentiment analysis can provide insights into how players feel about specific aspects of the game, such as changes introduced in a patch or the state of certain champions or items. When aggregated, these scores can help infer whether the community generally likes or dislikes the meta. However, this approach assumes that the sentiment expressed in posts accurately reflects broader player opinions, which may not always be the case.

One key consideration is the inherent bias in social media platforms like Reddit, where users are more likely to post or engage in discussions about things they find frustrating or want to complain about. This could lead to a disproportionate representation of negative sentiment, as players are often motivated to express dissatisfaction more than satisfaction. Consequently, sentiment scores might skew towards the negative, even if the broader player base is relatively content with the meta. Additionally, many posts in League of Legends subreddits are queries or discussions seeking advice on gameplay, which might not inherently reflect positive or negative sentiment but could still affect sentiment classification results. These factors highlight the need for caution when interpreting sentiment scores, as they may not fully capture the nuanced and diverse opinions of the entire player community.

6.5.2 Sentiment Classification Models

The first sentiment classification model used is the bert-base-multilingual-uncased-sentiment model [71] (hereafter referred to as "BERT base model"), a variant of the BERT model that has been fine-tuned specifically for sentiment analysis on product reviews across six languages: English, Dutch, German, French, Spanish, and Italian. This model predicts the sentiment of a review as a star rating, ranging from 1 to 5 stars. It was trained on a substantial dataset of product reviews, including 150,000 English reviews, 137,000 German reviews, 140,000 French reviews, 80,000 Dutch reviews, 72,000 Italian reviews, and 50,000 Spanish reviews. The model's accuracy varies across languages, achieving an exact match accuracy of 67% for English reviews and slightly lower exact match accuracy for other languages, such as 61% for German and 59% for French and Italian. However, when considering reviews where the predicted number of stars differs by only one from the actual rating (off-by-1 accuracy), the model performs exceptionally well, with accuracy

ranging from 93% to 95% across all languages. This high level of accuracy makes it a robust tool for sentiment analysis, even in contexts beyond product reviews, such as analyzing Reddit posts about League of Legends, though users should remain aware of the model's training background and potential biases.

The BERT base model is applied to compute sentiment scores for Reddit posts, focusing on both the title and content of each post. The process begins by loading the pre-trained tokenizer and model from the nlptown/bert-base-multilingual-uncased-sentiment repository [71], which are used to encode text and predict sentiment, respectively. A function is defined to take a text string as input, tokenize it, and pass it through the sentiment model to obtain a sentiment score. The function handles potential errors when attempting to shorten the text if an exception occurs, ensuring the process continues smoothly even with problematic text inputs. The sentiment score is determined by taking the index of the highest logit from the model's output, which corresponds to the predicted sentiment class (ranging from 0 to 4, where 0 is the worst sentiment and 4 is the best). The script then loads a dataset of Reddit posts and computes sentiment scores for the titles of these posts, storing the results in a new column called BERT-BASE Title Sentiment. It then proceeds to compute sentiment scores for the content of the posts, but only for self-posts. The sentiment scores for content are stored in another column, BERT-BASE Content Sentiment. This workflow adds BERT base sentiment scores to the Reddit posts dataset for further analysis.

Furthermore, the script performs sentiment classification on the Reddit comments dataset using the same classification model. It begins by loading the dataset of Reddit comments. To manage memory usage and processing time effectively, the dataset is divided into smaller chunks of 500 comments each. For each chunk, the sentiment score for the content of each comment is computed using the same function as applied in the posts dataset. This function, as explained earlier, tokenizes the comment text, processes it through the BERT model, and returns a sentiment score ranging from 0 (worst) to 4 (best). After processing each chunk, the results are appended to the DataFrame, which accumulates all processed comments and their sentiment scores.

The second sentiment classification model used is the Twitter-roBERTa-base model [14] (hereafter referred to as "roBERTa base model"), a roBERTa-base model specifically trained on approximately 124 million tweets collected between January 2018 and December 2021. This model has been fine-tuned

for sentiment analysis using the TweetEval benchmark, making it particularly well-suited for analyzing sentiment in English-language social media content. The model categorizes sentiments into three labels: 0 for Negative, 1 for Neutral, and 2 for Positive. Developed as part of the TimeLMs project [59] and integrated into the TweetNLP library [13], this model is optimized for understanding and analyzing sentiment in tweets and similar social media posts. Its specialization in social media language makes it a valuable tool for sentiment analysis in contexts like Reddit, where user-generated content shares similarities in tone and informal language.

The roBERTa base model is applied to compute sentiment scores for Reddit posts, similar to the process used with the BERT base model. The script begins by loading the tokenizer and model which are used to tokenize text and predict its sentiment, which is classified into three categories: 0 for negative, 1 for neutral, and 2 for positive sentiment. Another function is defined to process each text input, tokenize it, and pass it through the RoBERTa base model to obtain a sentiment score. Again, the function handles errors by truncating the text if necessary and returns the sentiment class with the highest probability from the model's output logits. The script then loads the existing dataset of Reddit posts, which already includes sentiment scores from the previous model. It computes sentiment scores for the titles of the posts using the RoBERTa model, storing these scores in a new column called RoBERTa Title Sentiment. Next, the script proceeds to calculate sentiment scores for the content of self-posts, storing these in another new column, RoBERTa Content Sentiment. The updated posts dataset, now contains sentiment scores from the RoBERTa model.

In a final step, the RoBERTa base model is applied to classify the sentiment of Reddit comments, again, following a process similar to what was done for Reddit posts. First, the existing dataset of Reddit comments is loaded, which already contains sentiment scores from the BERT base model. The dataset is divided into smaller chunks of 500 comments each and the script iterates over each chunk of comments, applying the defined RoBERTa base model function to the content of each comment to determine its sentiment score. The exact PyTorch [20] configuration used to employ the sentiment classification models can be inspected through Appendix H.

6.6 Web-Scraping Champion Meta

Aggregate League of Legends champion statistics, such as win, pick, and ban rates, are collected to inform the interpretation of which champions are pop-

ular or unpopular according to Reddit content. These statistics provide a quantitative measure of how champions are performing and being perceived in the game across different patches, offering valuable context when analyzing discussions on Reddit. By comparing these in-game statistics with sentiment and term frequency data from Reddit, a better understanding of the relationship between a champion's in-game performance and community perception can be established.

To gather this data, web scraping is employed to extract information from three prominent websites that provide comprehensive aggregate data for each champion across patches: LoLalytics, METAsrc, and U.GG. These platforms are known for their detailed and up-to-date statistics, which are based on large samples of games played at various levels of competition. By aggregating data from these sources, a more complete and accurate picture of champion performance and popularity is obtained.

6.6.1 LoLalytics

LoLalytics [58] is a popular online platform that provides detailed analytics and statistics for League of Legends, focusing on champion performance across various regions, ranks, and patches. It aggregates data from millions of games to offer insights into champion Win, Pick, and Ban rates, along with other metrics such as item builds, skill orders, and player performance trends. LoLalytics is widely used by players and analysts to track meta shifts, optimize gameplay strategies, and understand the current state of the game by offering a comprehensive view of how champions are performing in real-time.

6.6.2 METAsrc

METAsrc [68] is another online platform that provides comprehensive analytics and statistical data for League of Legends, helping players understand the current meta and optimize their gameplay. It aggregates data from millions of matches across various regions and ranks, offering detailed insights into champion performance, including Win, Pick, and Ban rates, as well as recommended builds, runes, and counters. METAsrc is known for its user-friendly interface and up-to-date information.

6.6.3 U.GG

U.GG [102] is a widely-used platform that provides in-depth analytics and statistics for a variety of games such as Valorant, World of Warcraft and

League of Legends. It offers players insights into League of Legends champion performance, meta trends, and optimal strategies. By gathering data from a vast number of games the site delivers accurate and real-time information on Win, Pick, and Ban rates, as well as recommended builds, runes, and counters for each champion. U.GG is favored by players for its intuitive interface and reliable data.

6.6.4 Data Scraping Methods

The scraping process is similar across all three websites, however, due to differences in the websites' HTML structures separate scrapers are developed for each platform. Generally, each scraper starts by scraping a list of all League of Legends champions from the specific website. An HTTP GET request is sent to the website's main League of Legends champion overview page. The HTML content of the page is then parsed using the BeautifulSoup library [83] with the "lxml" parser. It locates the HTML element containing the list of champions by finding a div element with specific classes. Within this container, it extracts all anchor tags, which contain the links to individual champion pages. For each champion, the code retrieves the champion's name from the URL and the full link to the champion's individual page. The extracted champion names and links are stored in a DataFrame with two columns: 'Name' for the champion's name and 'Link' for the URL to their page. In the next steps detailed statistics for each League of Legends champion from each respective website are collected, focusing on various patches, regions, and ranks. The scraper starts by defining dictionaries for rank, region, and patch criteria, which are used to construct different URLs for scraping specific configurations of champion data. These criteria include different ranks like "Gold" and "Diamond+", regions like "World", "NorthA", and "EUW", and patches from 14.4 to 14.8. Ideally, data from patches 14.1 to 14.3 would also be collected in the scope of this research approach, however, the websites limit the number of patches backwards for which data is accessible.

The scraper initializes an empty DataFrame, to store the collected data. This DataFrame includes columns for the champion's name, patch, region, rank, win rate, pick rate, ban rate, and other relevant metrics. It iterates over all champion links obtained earlier and applies the defined rank, region, and patch criteria to generate specific URLs for each combination. For each URL, the code sends a request to retrieve the HTML content of the page, parses it with BeautifulSoup, and extracts the champion's name and various statistics (e.g.: win rate, pick rate, ban rate, tier, and the number of matches). These

statistics are then stored in the DataFrame. Each iteration's collected data, including the URL and a unique sample ID, generated by converting a string combining the champion's name, rank, region, and patch into a hexadecimal format, is appended to the DataFrame. The results are periodically saved to a CSV file to ensure data is not lost during the scraping process. After each champion link collection, the code interrupts for one second to avoid taxing the server. The three collected datasets each contain win rate, pick rate, and ban rate data for 5010 different rank-region-patch configurations across 167 champions.

6.7 Data Dragon

Riot Games' Data Dragon [29] is an official data repository that provides comprehensive and up-to-date information about the various in-game elements in League of Legends, including champions, items, runes, and spells. It is regularly updated with each new patch, making it a useful resource for developers, analysts, and players who want access to the most accurate game data. Data Dragon offers detailed statistics for each champion, such as for example base attack damage, armor, movement speed, and other key attributes that define a champions strengths in the game. This data is particularly useful for comparing champion statistics across different patches. By analyzing the data provided by Data Dragon, changes in champion attributes can be tracked over time, which helps in understanding how balance changes (such as buffs or nerfs) affect champions. Comparing these actual in-game statistics with the aggregated performance metrics (like win rate, pick rate, and ban rate) from other sources like LoLalytics, METAsrc, and U.GG can provide deeper insights into how specific changes impact player behavior and champion effectiveness. This comparison can be helpful for contextualizing the impact of patch updates on champion performance and for validating trends observed in Reddit community discussions.

In order to retrieve and store these detailed champion statistics from Riot Games' Data Dragon for multiple patches a separate scraper is developed. It starts by sending an HTTP request to obtain a JSON file containing the list of all champions in the game as of patch 14.8. This JSON file is then parsed to extract the unique identifiers for each champion, which are stored in a list. The scraper then sets up column names for a DataFrame that will store various champion statistics, including basic stats (like attack damage and health), spell stats (like cooldowns, costs, and ranges for abilities Q, W, E, and R), champion classes (like Fighter, Mage), and additional details like the number of skins available for each champion. It also specifies the patch

versions of interest, ranging from 14.4 to 14.8, with each version mapped to its respective Data Dragon version string. The scraper then iterates over each champion ID and each patch version. For each combination, it constructs a URL to retrieve the champion's data for that specific patch. The data is then parsed to extract the various champion statistics and the timestamp of data collection. These details are stored in the DataFrame. The scraper also includes a one-second-pause after each iteration to avoid overwhelming the server. This scraper effectively compiles a dataset of champion statistics across multiple patches, which can be used for analyzing how champions' in-game attributes change over time and how these changes might correlate with performance metrics and community sentiment. In total it contains 835 rows of data for 167 champions across the selected patches.

7 Results & Analysis

In the following chapters, the results and comprehensive analysis of collected data related to the game League of Legends is presented, focusing on several key aspects across different patches and subreddits. Starting off, trends and changes in player discussions across are examined across various patches (Section 7.1), showcasing the changes in community sentiment and behavior. The analysis then extends to multiple League of Legends related subreddits, where the discourse across these forums is explored (Section 7.2) to identify patterns and shifts in community focus. Additionally, a detailed examination of unique (Section 7.3) and common (Section 7.4) TF-IDF terms is conducted. Here the specific language and topics are uncovered, that are distinct to certain patches or subreddits, as well as those that are consistently prevalent. Furthermore, items (Section 7.5) and champions (Section 7.6) that dominate discussions are identified, offering insights into the in-game elements that generate the most community engagement and debate. These analyses collectively aim to provide an understanding of the evolving dynamics within the League of Legends community across different patches and communities, and ultimately to showcase a possible approach to inform future game updates.

7.1 Analysis across Patches

In order to compare collected Reddit content across the patches 14.1 to 14.8, the posts dataset is grouped based on the "Patch" column. Initially focusing on titles of all posts, the mean values for the "Score" (i.e. the Reddit upvote score of the posts attached to the specific title), "BERT-BASE Title

Sentiment", and "RoBERTa Title Sentiment" are computed for each patch, and the total number of posts per patch is counted. The result is a new DataFrame (Table 2), which includes the patch identifier, the average post score, the average post title sentiment scores, and the number of posts associated with each patch.

Patch	Score	BERT-BASE Title Sentiment	RoBERTa Title Sentiment	Number of Posts
14.1	28.385195	2.271134	1.017099	9416
14.2	30.945408	2.292579	1.031703	7034
14.3	31.243228	2.273138	1.032590	7088
14.4	32.764566	2.295393	1.024051	5904
14.5	32.139158	2.252589	1.021717	5986
14.6	30.276269	2.304831	1.029443	5672
14.7	48.761316	2.190290	1.010337	6385
14.8	45.552228	2.179410	0.987207	6644

Table 2: Average Title Metrics and Total Number of Reddit Posts

To plot this data, the columns 'Score', 'BERT-BASE Title Sentiment', 'RoBERTa Title Sentiment', and 'Number of Posts' are standardized in the following manner. Let x be a value of a post title metric from Table 2, $\mu(x)$ the metric average across patches, and $\sigma(x)$ the metric standard deviation across patches. Then the standardized value is calculated according to Equation 1.

$$\frac{x - \mu(x)}{\sigma(x)} \tag{1}$$

In a next step all standardized metrics are offset using Equation 2, in order to shift negative values to become greater than zero. Here, x_s represents an already standardized metric from Table 2 and $\min(x_s)$ represents the smallest standardized metric.

$$x_s - \min(x_s) \tag{2}$$

Figure 8 displays a line plot showing the trends of these standardized metrics across different patches. The plot includes a separate line for each metric, with markers indicating values at each patch.

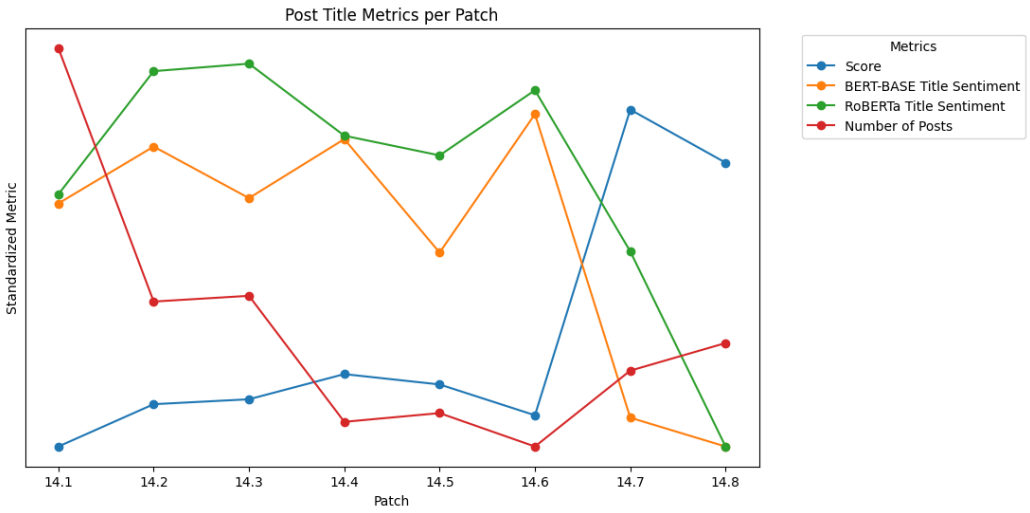


Figure 8: Standardized Post Title Metrics per Patch

The same process can be performed for metrics focusing on the posts contents rather than the titles. However, in this case, only self-posts can be considered since sentiment scores have not been evaluated for posts containing contents other than plain text (e.g.: images, videos, links). Table 2 lists the average post score, the average post content sentiment scores, and the number of self-posts associated with each patch.

Patch	Score	BERT-BASE Content Sentiment	RoBERTa Content Sentiment	Number of Self-Posts
14.1	10.613057	1.826079	0.920028	5652
14.2	10.866148	1.775875	0.900389	3855
14.3	12.125247	1.826087	0.909091	4048
14.4	11.580826	1.807670	0.912389	3390
14.5	12.087209	1.782849	0.919186	3440
14.6	12.512548	1.790541	0.895109	3108
14.7	31.488736	1.759608	0.874371	3773
14.8	29.922027	1.746653	0.883697	3809

Table 3: Average Content Metrics and Total Number of Reddit Self-Posts

The plot derrived from Table 2 and displayed in Figure 9 shows the standardized post content metrics across different patches.

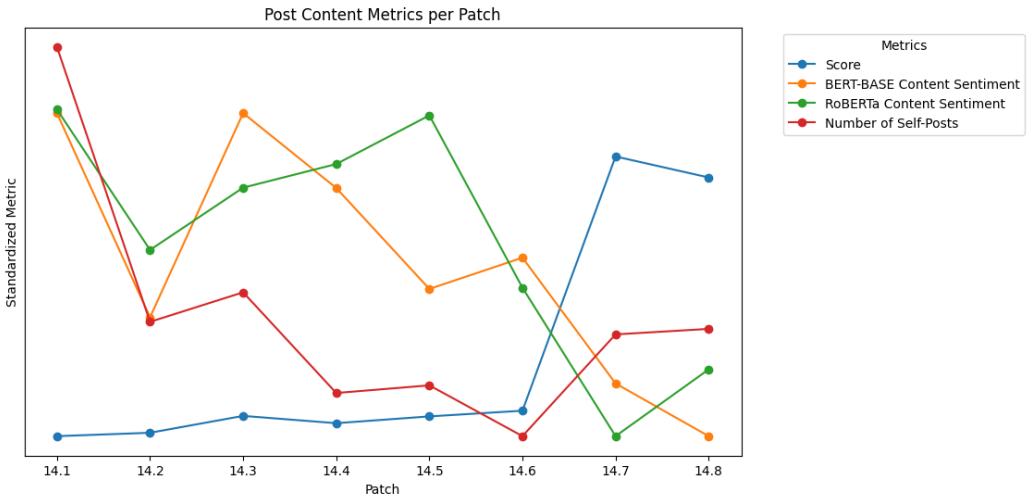


Figure 9: Standardized Post Content Metrics per Patch

Regarding the comments dataset, metrics can be computed in a similar fashion. Since comments only contain a content body without a title, metrics are only computed for the contents.

Patch	Score	BERT-BASE Comment Sentiment	RoBERTa Comment Sentiment	Number of Comments
14.1	4.790050	1.951648	0.966912	49988
14.2	4.854921	1.942875	0.958075	39475
14.3	4.945291	1.989524	0.971840	41620
14.4	4.934512	1.974870	0.962031	34739
14.5	5.065768	1.958613	0.959240	35108
14.6	5.140830	1.958912	0.950494	32905
14.7	12.536859	1.886210	0.903474	47614
14.8	11.356284	1.868616	0.888810	47639

Table 4: Average Comment Metrics and Total Number of Reddit Comments

Analogously, the plot derived from Table 4 and displayed in Figure 10 shows the standardized comment metrics across different patches.

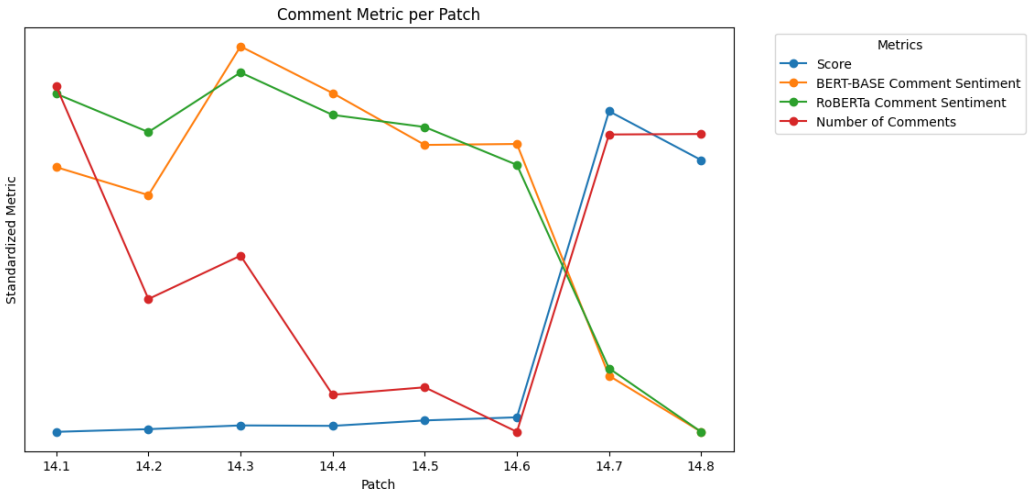


Figure 10: Standardized Post Content Metrics per Patch

The plots in Figures 8, 9, 10 reveal several key trends in the data. The number of posts and comments per patch is initially high at the start of the season (patch 14.1), which possibly reflects the community's heightened engagement due to substantial game changes introduced at the beginning of the new season, such as map reworks, item changes, and other meta shifts. However, this initial activity drops off quickly in subsequent patches. Interestingly, while the upvote scores of posts and comments remain relatively stable early in the season, they show a noticeable increase in patch 14.7. This rise in upvote scores, despite a less pronounced increase in the number of posts and comments, may suggest that the community strongly agreed with the sentiments expressed during this patch. The sentiment scores derived from BERT base and RoBERTa models for post titles and contents only vaguely correlate, yet both exhibit a decrease in sentiment starting from patch 14.6, with the only exception for RoBERTa content sentiment in patch 14.8. The sentiment scores for comments, however, show a closer correlation and similarly decline after patch 14.6. Assuming these sentiment scores are a proxy for the community's feelings about the League of Legends meta, this dip in sentiment indicates growing dissatisfaction. Furthermore, if the assumption is made that the upvote score serves as a proxy for community agreement, the spike in upvotes during patch 14.7, combined with the decrease in sentiment, suggests that the League of Legends Reddit community was largely unified in its discontent.

One possible explanation for this noticeable dip in sentiment and concurrent rise in upvotes and post activity around patch 14.7 could be the Visual

and Gameplay Update (VGU) for the champion "Skarner". This update, which took place during the 14.7 patch [86], might have significantly impacted the meta, making the champion overwhelmingly strong, which could have led to widespread complaints and dissatisfaction within the community. The strong agreement across posts, as reflected in the upvotes, possibly indicates that many players shared these negative sentiments about the changes, contributing to the observed trends in the data.

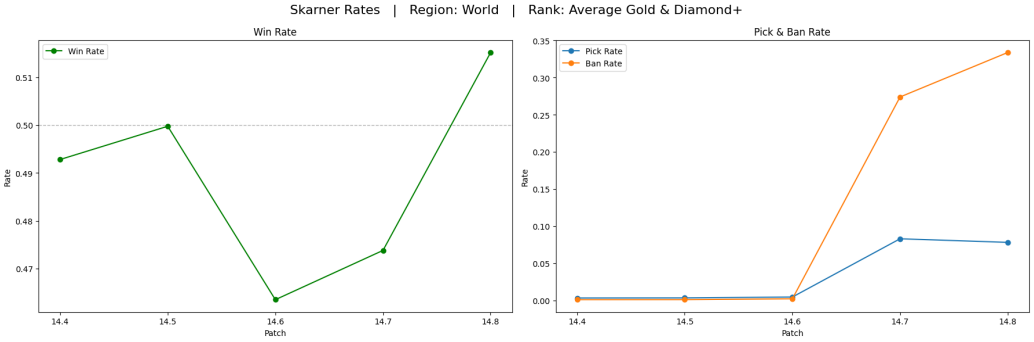


Figure 11: Skarner Win/Pick/Ban Rates across Patches

Indeed, when examining the data collected from Riot Games' Data Dragon repository (Figure 12), one can see that Skarner received significant buffs in patch 14.7, particularly to his spell abilities. These changes likely made Skarner much more effective in gameplay, contributing to his increased dominance in matches. At the same time, when inspecting the aggregate champion data (collected through the web scrapes of LoLalytics, METAsrc, and U.GG, then averaged out) in Figure 11, one can observe a noticeable increase in Skarner's pick and ban rates following the VGU. Prior to this update, Skarner's pick and ban rates had remained consistently low, reflecting his relatively minor role in the meta. However, the substantial buffs introduced in patch 14.7 appear to have shifted the community's perception, leading to a surge in his popularity and also possibly driving the heightened activity and discussions seen in the corresponding Reddit data. This alignment between the in-game changes and the community's reaction offers a possible explanation that the VGU not only impacted Skarner's viability but also sparked significant discourse and agreement among players, as reflected in both sentiment scores and upvote trends.

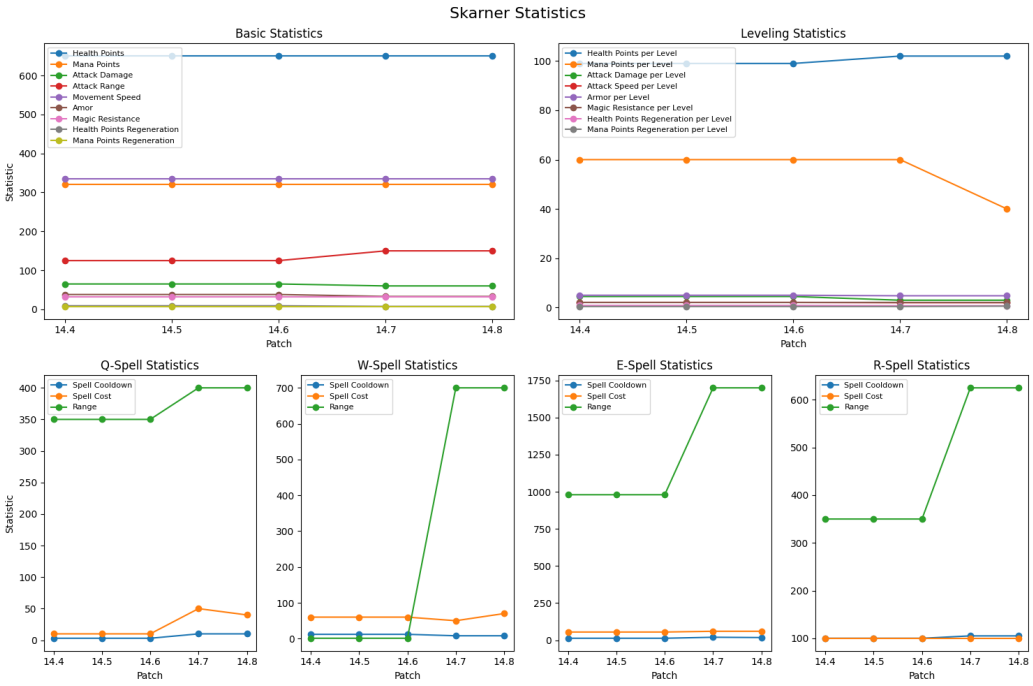


Figure 12: Skarner Champion Statistics across Patches

7.2 Analysis across Subreddits

In order to compare computed metrics across different subreddits, several operations on a dataset of Reddit posts are performed. First, focusing on post titles, the posts are grouped by their corresponding subreddits and for each subreddit group the mean values for the upvote score, BERT-BASE Title Sentiment, and RoBERTa Title Sentiment are calculated. Additionally, the number of posts in each subreddit are counted. The result is stored in a new DataFrame. Table 5 lists the various average title-based metrics for the top 5 subreddits based on average upvote score in the first 5 rows, and for the bottom 5 subreddits throughout rows 6-10.

The standardized average upvote scores and number of posts of the subreddits from the two subsets in Table 5 are plotted in Figure 13.

Subreddit	Score	BERT-BASE Title Sentiment	RoBERTa Title Sentiment	Number of Posts
leagueoflegends	141.470618	1.876900	0.848531	1974
AhriMains	104.588333	2.971667	1.120000	600
ornnmains	92.750877	2.189474	1.014035	285
GwenMains	92.152961	2.467105	1.090461	608
Kindred	81.957447	2.338652	1.097518	564
fizzmains	6.925532	2.329787	1.005319	188
MaokaiMains	6.892857	2.303571	1.044643	112
XinZhaoMains	6.831579	2.263158	1.084211	95
LeagueofLegendsMeta	1.812500	2.687500	0.937500	32
MarksmanMains	0.821429	3.535714	1.214286	28

Table 5: Average Title Metrics and Total Number of Reddit Posts for Top/Bottom 5 Subreddits based on Average Score

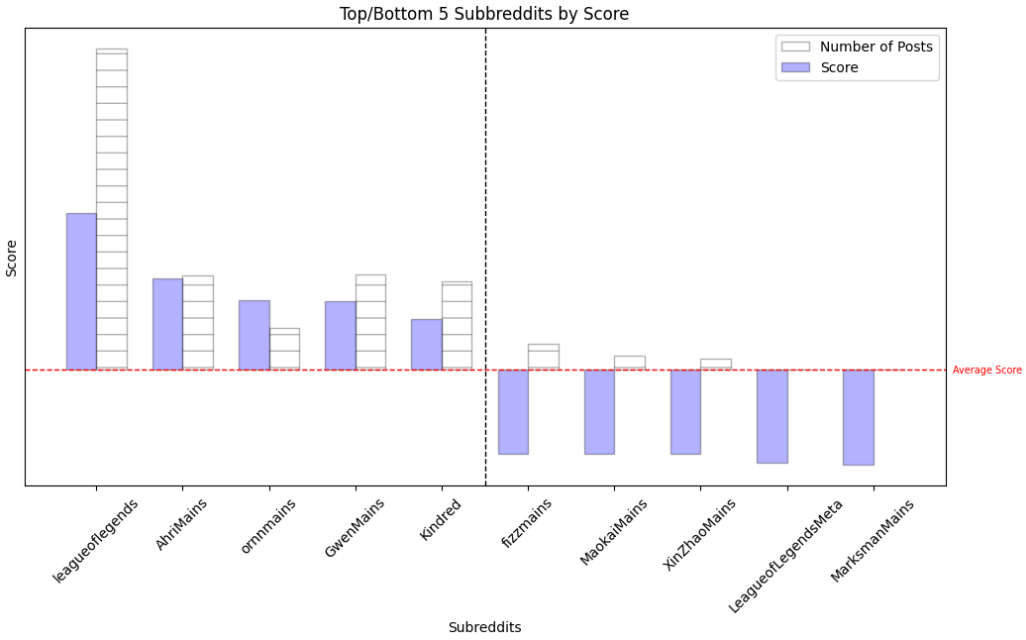


Figure 13: Top/Bottom 5 Subbreddits by Score

The size of the purple bars represents the average upvote score of the given subreddit oriented around the average score of all subreddites as a baseline. Bars plotted below the average score baseline are not necessarily negative but merely below average. For reference, the sizes of the hatched grey bars represent the number of posts for each given subreddit however the

orientation of the bars is not indicative.

Subreddit	Score	BERT-BASE Title Sentiment	RoBERTa Title Sentiment	Number of Posts
MarksmanMains	0.821429	3.535714	1.214286	28
AhriMains	104.588333	2.971667	1.120000	600
MorganaMains	73.643443	2.844262	1.122951	244
BraumMains	18.989011	2.780220	1.197802	91
RakanMains	30.628571	2.771429	1.238095	210
MalzaharMains	15.393064	1.872832	0.971098	173
reksaimains	13.374179	1.857768	0.879650	457
malphitemains	7.008130	1.691057	0.967480	123
nunumains	12.237037	1.674074	1.059259	135
LeaguePBE	8.958716	0.779817	0.733945	218

Table 6: Average Title Metrics and Total Number of Reddit Posts for Top/Bottom 5 Subreddits based on Average BERT-BASE Title Sentiment

In order to receive a more detailed view of sentiment across different subreddits, the original grouped DataFrame is sorted by the BERT-BASE Title Sentiment in descending order to identify the subreddits with the highest and lowest average sentiment scores. The top five and bottom five subreddits are then selected, combined into a new DataFrame which can be seen in Table 6. Similar to before, the standardized average upvote scores and number of posts of the subreddits from the two subsets in Table 6 are plotted in Figure 14. The results of repeating the same process for the RoBERTa Title Sentiment scores can be seen in Table 7 and Figure 15. The analogously resulting plot figures from sorting by post content and comment data metrics can be inspected via Appendix I and Appendix J.

The Figures 13, 14, and 15 include dedicated grey hatched bars that represent the number of posts for each subreddit, thereby providing visual context for the activity level within each subreddit. However, the ranking of the subreddits in these figures is based solely on a single metric, such as the average score or sentiment, without considering the volume of posts. To achieve a more meaningful ranking, the subreddits are sorted by a metric that has been weighted by the number of posts in each subreddit. This approach accounts for both the quality and quantity of posts, offering a more comprehensive view of each subreddit's influence and activity. To accomplish this, again first focusing on titles, Reddit posts are grouped by their subreddits and the mean values for Score, BERT-BASE Title Sentiment, and RoBERTa Title Sentiment are calculated, while also counting the number of posts in

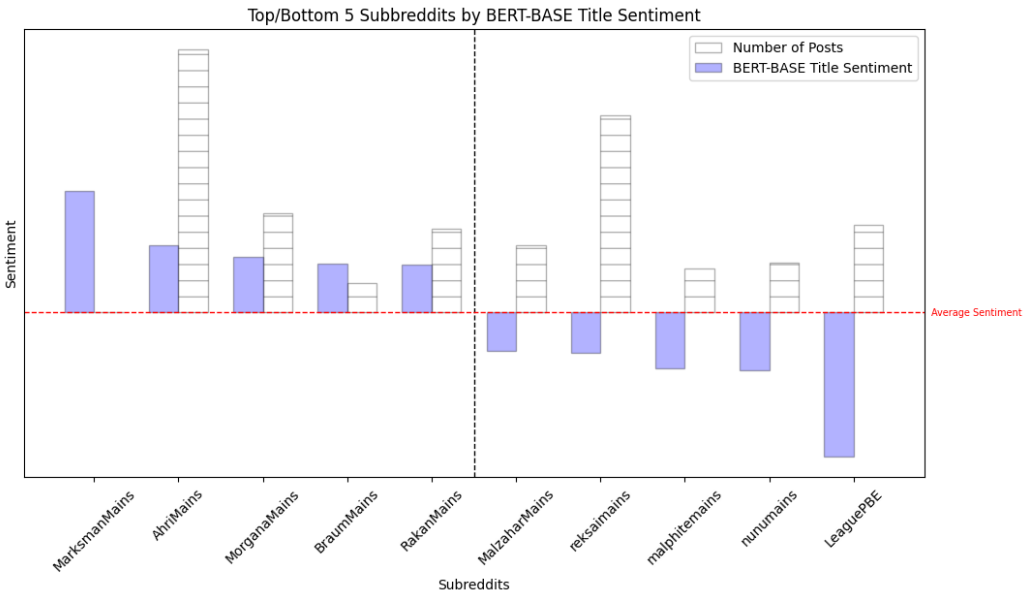


Figure 14: Top/Bottom 5 Subbreddits by BERT-BASE Title Sentiment

Subreddit	Score	BERT-BASE Title Sentiment	RoBERTa Title Sentiment	Number of Posts
RakanMains	30.628571	2.771429	1.238095	210
MarksmanMains	0.821429	3.535714	1.214286	28
BraumMains	18.989011	2.780220	1.197802	91
MilioMains	58.229412	2.376471	1.188235	170
NautilusMains	12.402778	2.500000	1.180556	72
Draven	63.572340	1.961702	0.904255	470
reksaimains	13.374179	1.857768	0.879650	457
summonerschool	26.129332	2.168216	0.863060	1183
leagueoflegends	141.470618	1.876900	0.848531	1974
LeaguePBE	8.958716	0.779817	0.733945	218

Table 7: Average Title Metrics and Total Number of Reddit Posts for Top/Bottom 5 Subreddits based on Average RoBERTa Title Sentiment

each subreddit. Then the values in these columns are standardized, and values in the column representing the number of posts is offset by subtracting its minimum value. A new column, Score Rank, is created by multiplying the standardized score by the number of posts, which ranks subreddits based on a combined measure of score and activity. The DataFrame is then sorted by the computed score rank in descending order, and the top and bottom

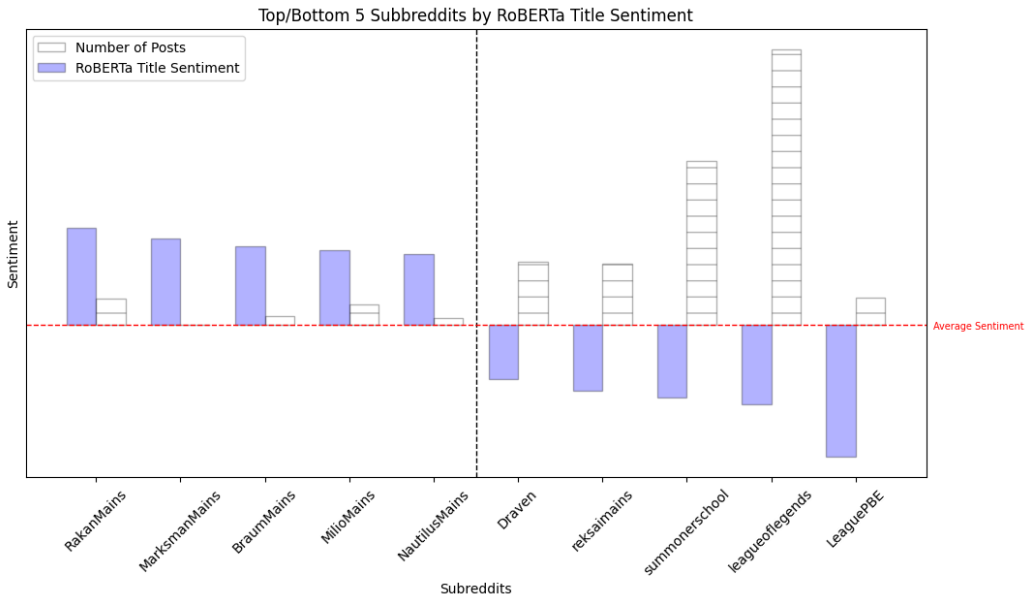


Figure 15: Top/Bottom 5 Subbreddits by RoBERTa Title Sentiment

five subreddits are selected to form a subset for visualization. Figure 16 displays the top and bottom five subreddits based on the computed score rank. Hatched bars representing the sentiment scores are additionally displayed for reference purposes and are aligned on the average score baseline which also represents the average for each sentiment score. Again, the number of posts are plotted using grey hatched bars which have been offset to be placed about the average score baseline with orientation of the bars being non-indicative.

Similarly, subreddits can by analysed and visualized by calculating a weighted sentiment score. Here, Reddit posts are grouped by subreddit and the mean scores and post counts are computed. These values are then standardized, and the number of posts is further offset. The sentiment rank, is then calculated as the average of the BERT-BASE and RoBERTa title sentiment scores and weighted by the number of posts in each subreddit. This weighted sentiment rank provides a meaningful measure of overall sentiment by accounting for both the sentiment quality and the volume of posts. The subreddits are then sorted based on the computed sentiment rank, and the top five and bottom five subreddits are selected for visualization. Figure 17 displays the top and bottom five subreddits based on the computed sentiment rank. The blue hatched bars representing the upvote scores are additionally displayed for reference purposes and are aligned on the average

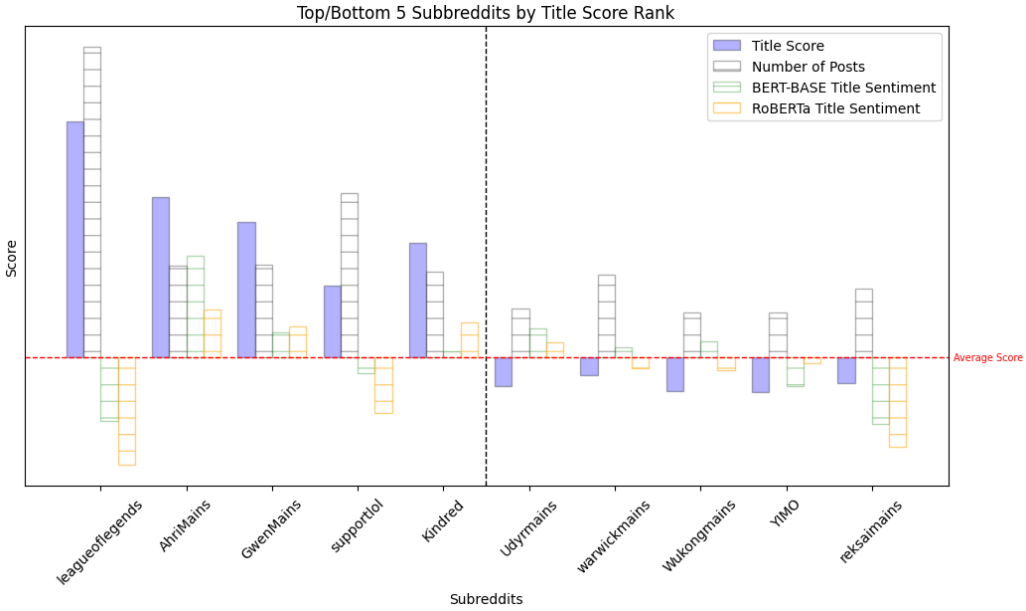


Figure 16: Top/Bottom 5 Subbreddits by Title Score Rank

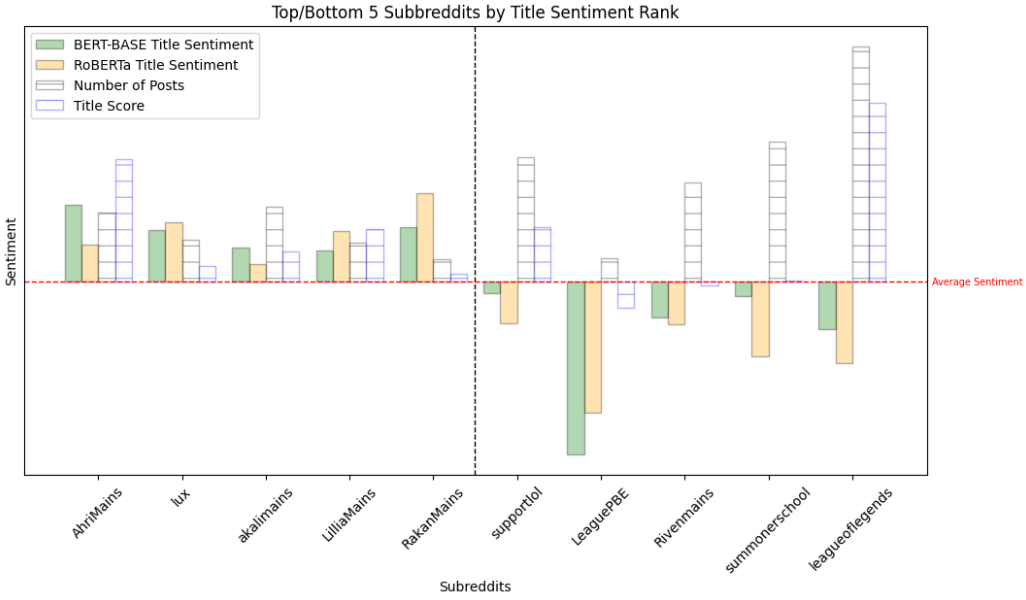


Figure 17: Top/Bottom 5 Subbreddits by Title Sentiment Rank

sentiment baseline which also represents the average baseline for each sentiment score. Again, the orientation of the grey hatched bars representing the number of posts per subreddit is not indicative due to offset. The analo-

gously resulting plot figures from sorting post content and comment data by score and sentiment ranks can be inspected via Appendix K and Appendix L.

It may also be valuable to analyze subreddits based on how volatile their upvote scores and sentiments are across patches. Understanding the volatility of these metrics can reveal how consistently the community's opinions and reactions shift in response to game updates and changes. High volatility in upvote scores might indicate that a subreddit has strong, varying opinions that fluctuate depending on the content and relevance of each patch. Similarly, volatility in sentiment scores could point to a community that is highly sensitive to changes in the game, where emotions and opinions rapidly shift in response to perceived improvements or nerfs in gameplay. In general, this type of analysis could help to identify subreddits that are more prone to extreme shifts in opinion, as well as those that maintain a more stable outlook. In order to rank subreddits based on their upvote score volatility across patches Reddit self-post are grouped by both subreddit and patch, computing the mean score and the number of posts for each combination. Then differences in scores between consecutive patches are computed. These differences are squared to turn each term positive, and the squared differences are then standardized. The standardized squared differences are added up across all patches for each subreddit, and weighed by the total number of posts to compute a score volatility rank. Subreddits are sorted by this score volatility rank, and the top five most volatile subreddits are selected for visualization. Figure 18 displays the scores of the top five subreddits based on the computed score volatility scores, across all patches. Analogously, in Figure 19 subreddits are sorted by sentiment volatility rank and the top five subreddits are plotted. Note that the mean of the BERT BASE Title Sentiment, RoBERTa Title Sentiment, BERT BASE Content Sentiment, and RoBERTa Content Sentiment is used to derive the sentiment volatility rank. The bottom five subreddits by score volatility rank and sentiment volatility rank can be observed in Appendix M.

In the across-subreddit analysis, the subreddit "leagueoflegends" appears frequently among the top or bottom five subreddits in all score, sentiment, and volatility rankings (except when ranked by RoBERTa content sentiment and BERT base title sentiment). A possible reason for this prominence could be the disproportionately large number of posts collected from this particular subreddit (as detailed in Appendix C), which likely amplifies its influence in the rankings. Another notable observation is the frequent appearance of the subreddit "Rivenmains", a community focused on the champion "Riven". This subreddit consistently ranks among the bottom five in title sentiment

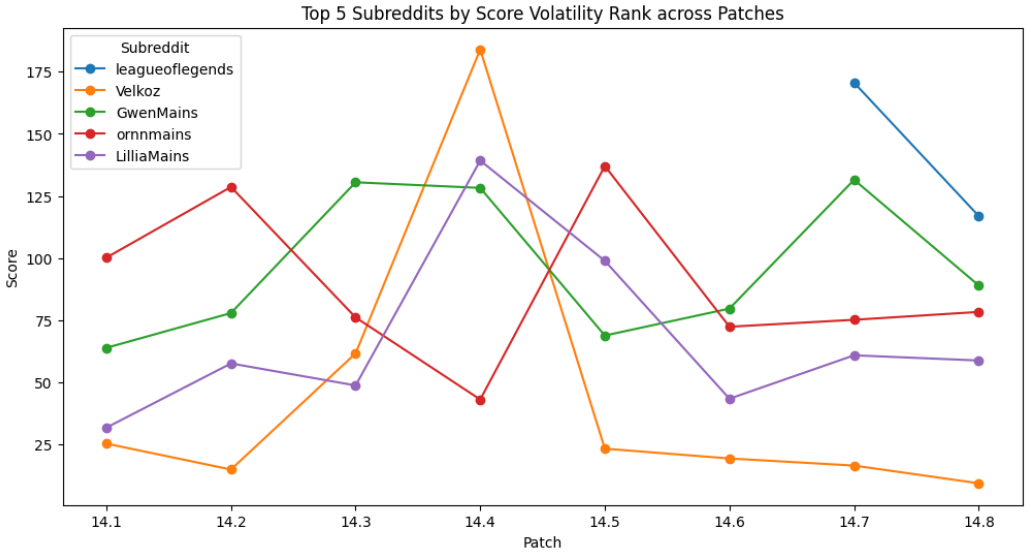


Figure 18: Top 5 Subbreddits by Score Volatility Rank

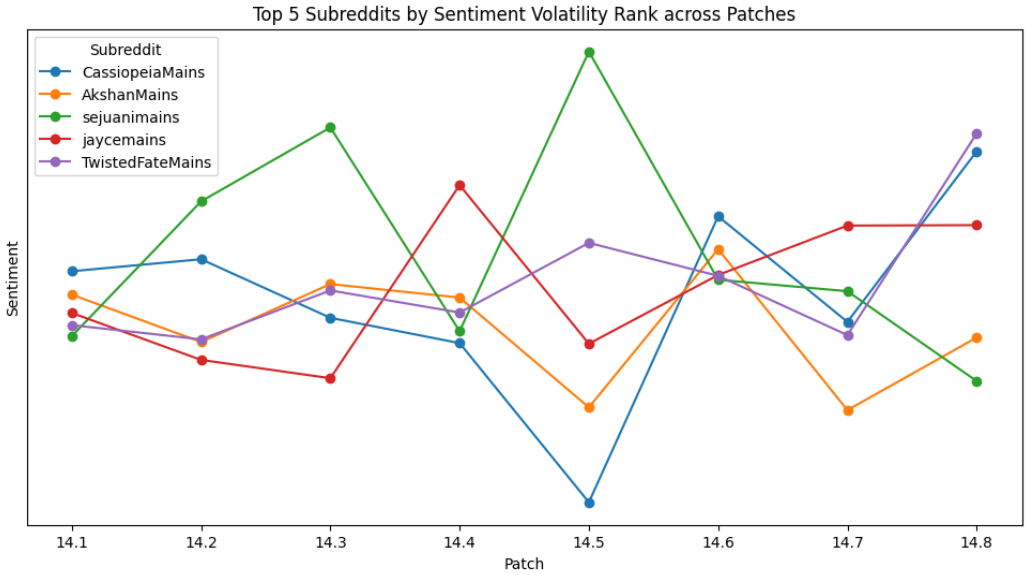


Figure 19: Top 5 Subbreddits by Sentiment Volatility Rank

rank, content sentiment rank, and comment sentiment rank, as well as in score volatility rank and sentiment volatility rank. This pattern may suggest that players active in the "Rivenmains" subreddit are consistently unhappy with the state of Riven's performance as a champion in League of Legends. This possibly points out a community that may feel overlooked or dissatisfied.

7.3 Analysis of Unique TF-IDF Terms

As detailed in Section 6.3.3, terms obtained by TF-IDF term clustering are filtered for uniqueness and then mapped to patches accordingly (Table 1). This results in a collection of unique terms specific to each patch and aims to provide a view of distinct vocabulary or themes used by Reddit communities during each game patch. In order to target the identified terms within the corpus of Reddit posts, first, title and content of each Reddit post are concatenated, and then the presence of unique terms is searched for within the combined texts. A function is defined and applied to find and extract these terms, after which the processed data is grouped by the identified unique terms, calculating average sentiment scores and other metrics, such as post upvote scores and comment counts, for each term. Lastly, a subset of the data based on terms associated with patch 14.1 is generated for further analysis and visualization.

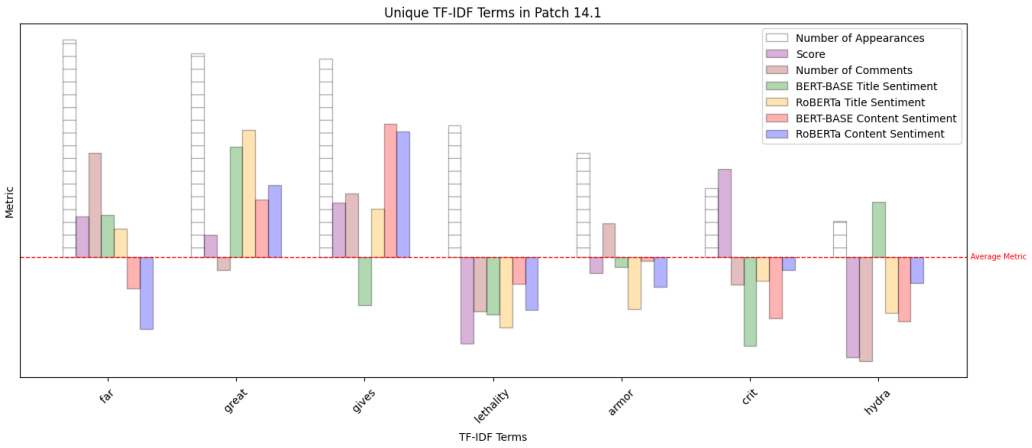


Figure 20: Upvote Scores, Comment Counts, and Sentiment Scores for Reddit Posts containing Unique TF-IDF Terms in Patch 14.1

Figure 20 displays the standardized upvote scores, comment counts, and sentiment scores for Reddit posts containing unique TF-IDF terms in the title or content body (or both), and that have been posted during Patch 14.1. Each colored bar, representing a specific metric is plotted with an orientation around the average metric baseline to indicate above or below average metric values. The average metric baseline serves as an indicator of the total average metric value of each individual metric. The grey hatched bars represent the number of posts which contain the given TF-IDF term, and are offset to be displayed for reference above the average metric baseline, making their orientation non-indicative. Plots generated through a similar process

as described above, for the unique TF-IDF terms identified for patches 14.2 to 14.8, can be observed in Appendix N.

Upvote scores and comment counts appear consistently either both above or both below average for all terms, with the exception of the term "shield" in patch 14.6. Terms describing game concepts with below average sentiment across all sentiment metrics include "lethality", "armor", and "crit" in patch 14.1, "buff" in patch 14.2, "eqeq" in patch 14.5, and "bot" in patch 14.8. In League of Legends, "lethality" refers to a specific kind of armor penetration, "armor" describes a defensive champion stat, "crit" refers to critical strikes that deal extra damage, "buff" signifies enhancing a champion's abilities through patches, "eqeq" refers to alternately casting of a champions E-spell and Q-spell, and "bot" commonly refers to the bottom lane or champions conventionally played in the bottom lane (however, "bot" is also occasionally used to describe players as AI controlled NPCs with limited in-game skill). Conversely, terms such as "tips" in patch 14.3 and "shield" in patch 14.6 have above-average sentiment scores, with "tips" indicating helpful advice, and "shield" referring to abilities or items that absorb damage. These findings might imply that players tend to react more negatively towards a certain selection of core gameplay mechanics, such as damage-related concepts like "lethality" and "crit", while more positive reactions are reserved for support-oriented terms like "shield" or general helpful advice.

7.4 Analysis of Common TF-IDF Terms

Further analysis can be conducted based on the terms shown in Section 6.3.4, which are identified during TF-IDF term clustering and are common across all patches. To this end, occurrences of these terms are searched for within each post. Then posts are grouped by the found terms and aggregate statistics are calculated, including the mean score, sentiment values, and comment count, while also counting how many times each term appears. The data is then standardized, and rankings based on the volume of posts containing a given term are computed. This is done for the post scores, comment counts, and average sentiment values. The top 5 and bottom 5 terms are selected based on their score ranks, comment ranks, and sentiment ranks, and are visualized in a Figures 21, 22, and 23.

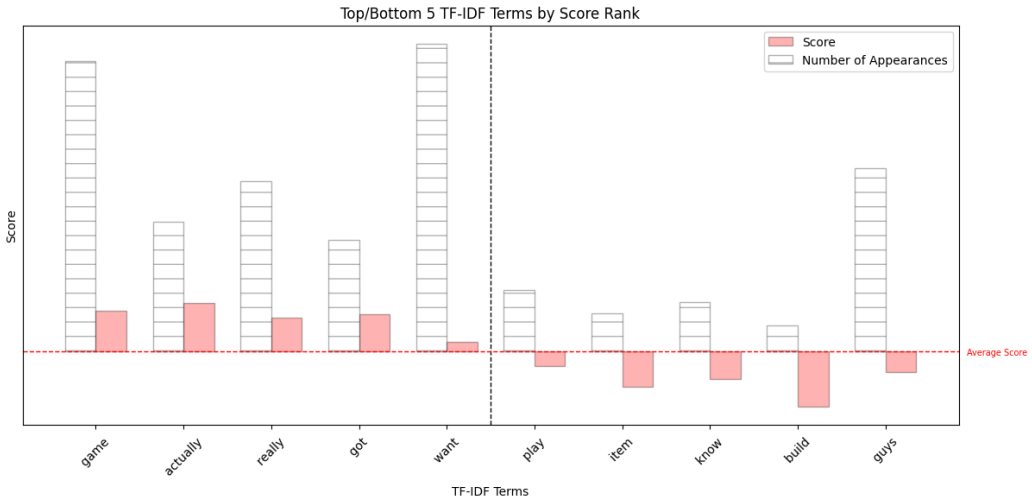


Figure 21: Top/Bottom 5 Common TF-IDF Terms by Score Rank

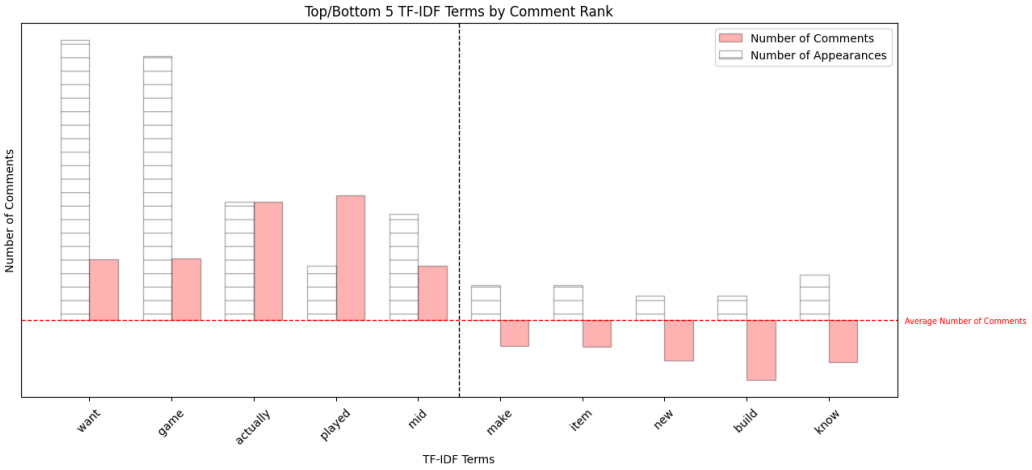


Figure 22: Top/Bottom 5 Common TF-IDF Terms by Comment Rank

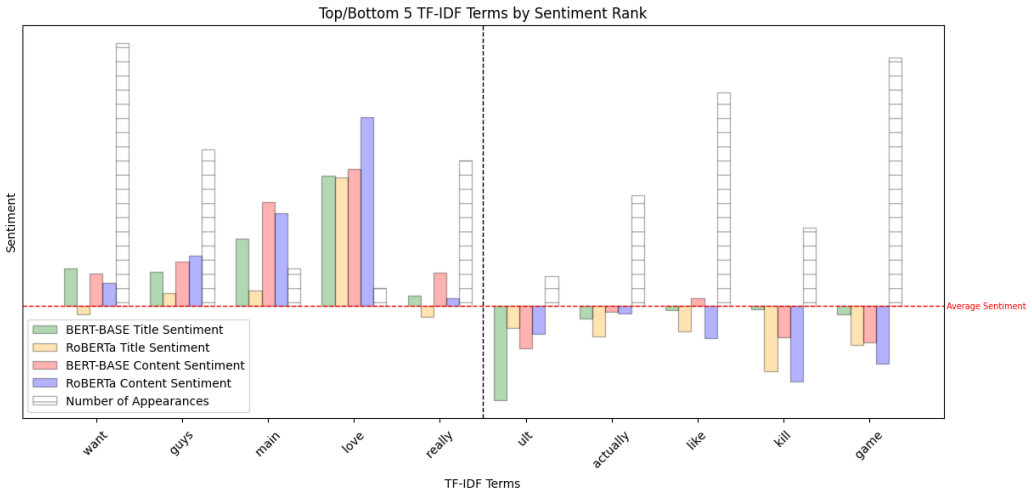


Figure 23: Top/Bottom 5 Common TF-IDF Terms by Sentiment Rank

The colored bars in Figures 21, 22, and 23 represent selected metrics that are plotted orienting around the average score baseline. The hatched grey bars represent the number of posts which contain the given term with non-indicative orientation. In order to compute the sentiment rank for the visualizations in Figure 23, the mean of the BERT BASE Title Sentiment, RoBERTa Title Sentiment, BERT BASE Content Sentiment, and RoBERTa Content Sentiment is used.

It may also be useful to investigate how much the scores and sentiment of posts containing common TF-IDF terms change across patches. This can provide insights into how player reactions evolve over time in response to specific game updates. Examining the volatility of these metrics can help identify terms or concepts that consistently provoke strong reactions, either positive or negative, across different patches. Investigating volatility captures fluctuations in player sentiment and engagement and may offer a clearer understanding of how certain gameplay changes or discussions affect the community mood. For this purpose, different metric volatility ranks of common TF-IDF terms are computed across game patches. First Reddit posts are grouped by term and patch, and the average score and the number of posts for each combination are calculated. In the next step, the score differences between consecutive patches for each term are calculated. These differences are squared to eliminated negative terms, and the squared differences are standardized. The volatility of each term is determined by adding up the squared differences and weighting them by the total number of term appearances. The top five most volatile terms regarding upvote score are identified

and visualized in Figure 24. Similarly the top five most volatile terms regarding sentiment score can be evaluated (Figure 25), by using the mean of the BERT BASE Title Sentiment, RoBERTa Title Sentiment, BERT BASE Content Sentiment, and RoBERTa Content Sentiment in order to derive a single sentiment metric to be weighted. Visualizations detailing the bottom 5 common TF-IDF terms by score and sentiment volatility rank across patches can be inspected via Appendix O.

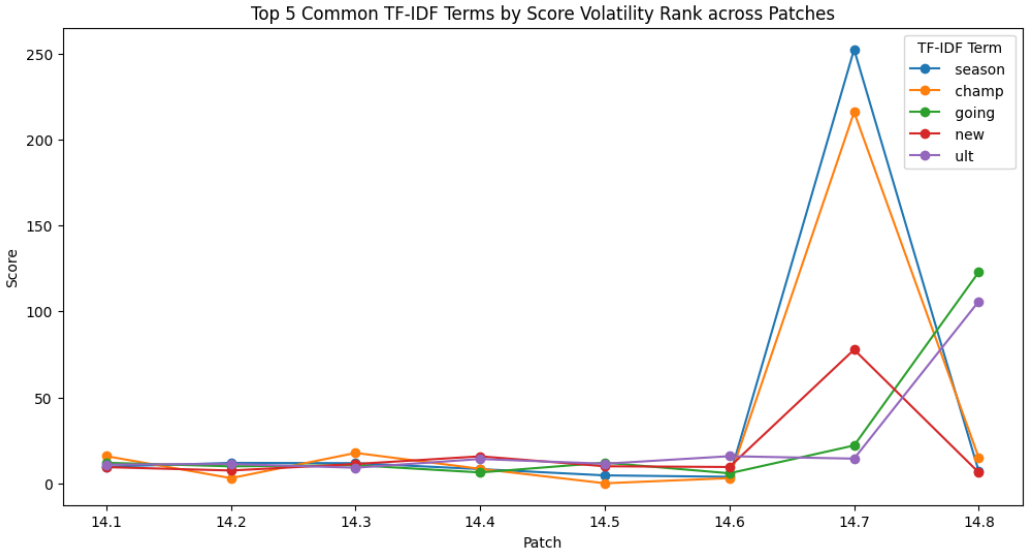


Figure 24: Top/Bottom 5 Common TF-IDF Terms by Score Volatility Rank

Observing the score volatility, one can see that the terms "season", "champ", and "new" display a noticeable increase in upvote scores during patch 14.7, suggesting heightened community interest or agreement during that time. However, looking at the bottom terms regarding score and sentiment volatility, the term "game" stands out with a noticeable increase in score but a simultaneous decrease in sentiment during patch 14.7. Interestingly, "game" ranks among the top five terms by score and comment rank but is the lowest-ranked in terms of sentiment rank, indicating a dissatisfaction with the overall gameplay. This trend likely reflects player frustration with broader game mechanics or updates, particularly during and after patch 14.7, which could be tied to the controversial Skarner VGU, as discussed in Section 7.1. The Skarner VGU comprises a significant overhaul of the champion Skarner in League of Legends and is aimed at modernizing his kit and improving his overall gameplay experience. The 14.7 patch focuses on making Skarner more engaging by reworking his abilities, and improving his visuals [86]. While the

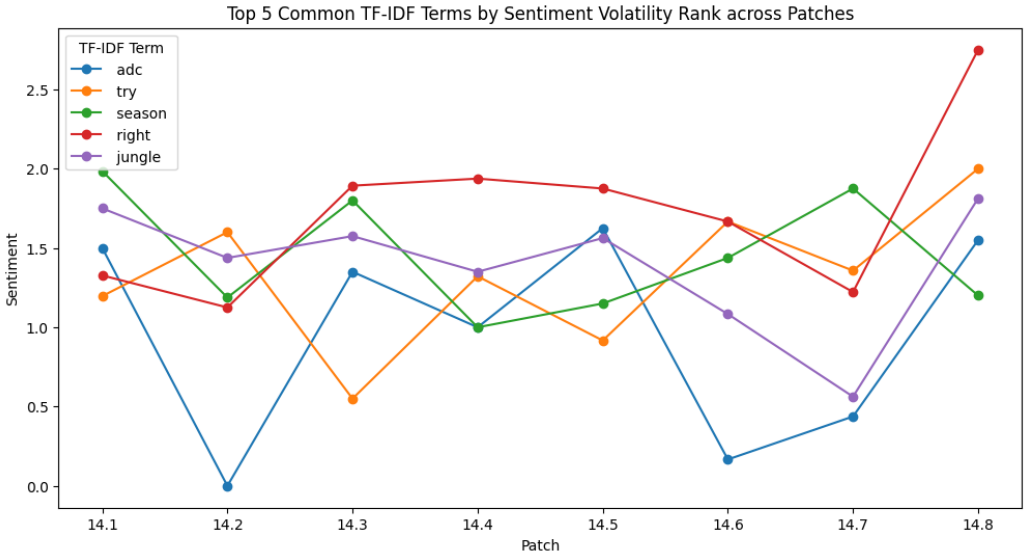


Figure 25: Top/Bottom 5 Common TF-IDF Terms by Sentiment Volatility Rank

changes aim to make him more relevant and appealing to players, the VGU evidently sparked mixed reactions within the community [100], with some players appreciating the update while others expressed frustration.

7.5 Analysis of discussion-dominating Items

The League of legends item dictionary, set up and introduced in Section 6.4.1, can be utilized to target posts containing the names of items. To do this, title and content of each collected Reddit post is concatenated, and the combined texts are filtered for items using the item dictionary. This way specific items discussed in the posts are identified. After this, statistics such as average sentiment and comment count per item are calculated, excluding any posts where no item names or aliases were found. Lastly, the top and bottom 5 items based on the total number of post they appear in are extracted visualized in Figure 26.

In order to rank items based on score and sentiment metrics weighted by the volume of posts the given item appears in, first, several metrics, including sentiment scores, comment counts, and the number of appearances are standardized. An additional offset is applied subtracting the minimum value of all post counts to the total number of item appearances. Then new ranking metrics are created based on weighted scores, comment counts, and

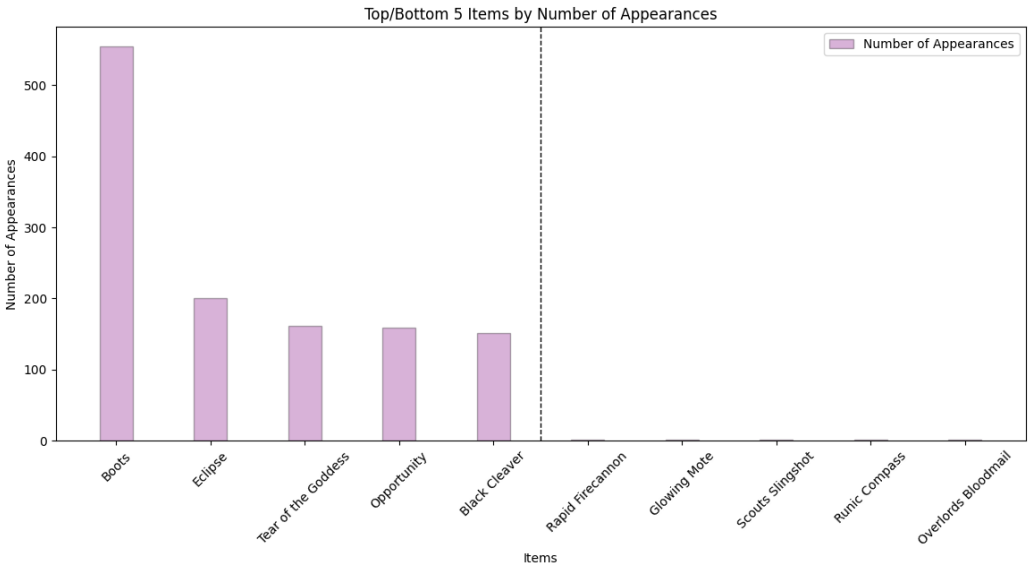


Figure 26: Top/Bottom 5 Items by Number of Appearance

aggregated sentiment, all scaled by the number of item post appearances. Figures 27, 28, and 29 display the top and bottom 5 items based on score, comment, and sentiment rank.

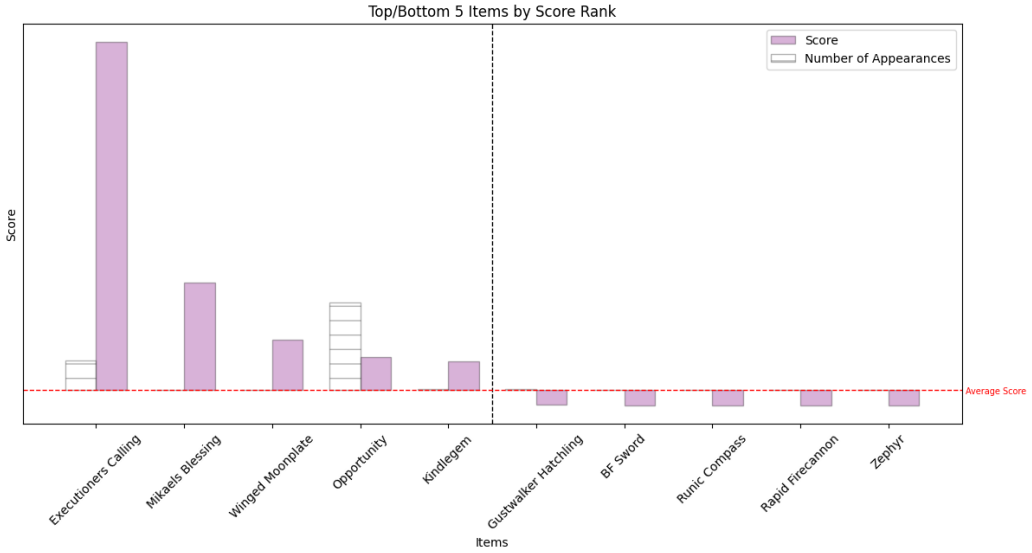


Figure 27: Top/Bottom 5 Items by Score Rank

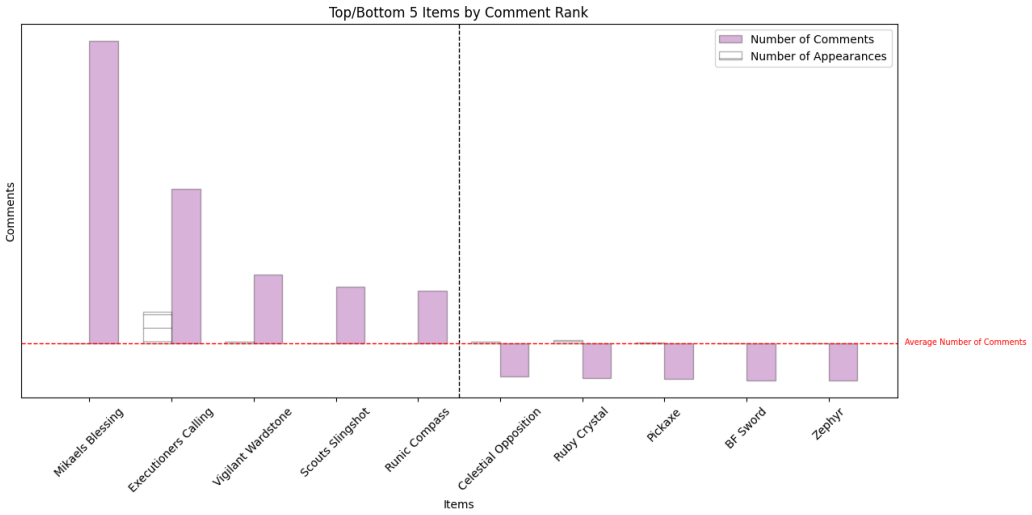


Figure 28: Top/Bottom 5 Items by Comment Rank

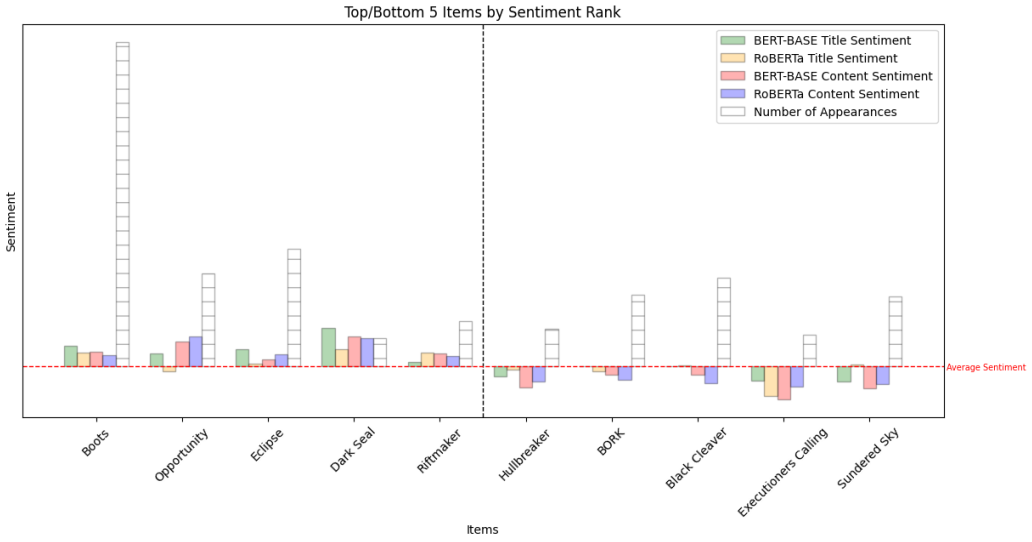


Figure 29: Top/Bottom 5 Items by Sentiment Rank

In order to visualize the volatility of the number of posts for different items across different patches posts are grouped by item and patch to compute the average score and the total number of posts per patch for each item. Then the squared difference in the number of posts between consecutive patches are calculated for each item to capture volatility. The squared differences are standardized, and the items are ranked by their sum of squared differences to identify the top five most volatile items by appearance (Figure 30). Next,

the focus can be shifted from the number of appearances to the item score volatility across patches. Instead of calculating the volatility based on the differences in the number of item posts, volatility is computed by looking at the changes in the upvote score. The differences in scores between consecutive patches are calculated and squared to capture volatility, which is then standardized. The final volatility ranking now multiplies the standardized squared score differences by the number of posts for each item. Figure 31 now displays the top five most volatile items according to score rank across patches. Similarly, rankings based on sentiment volatility rank are computed (Figure 32). The bottom 5 items based on appearance volatility, score volatility rank, and sentiment volatility rank can be inspected through Appendix P.

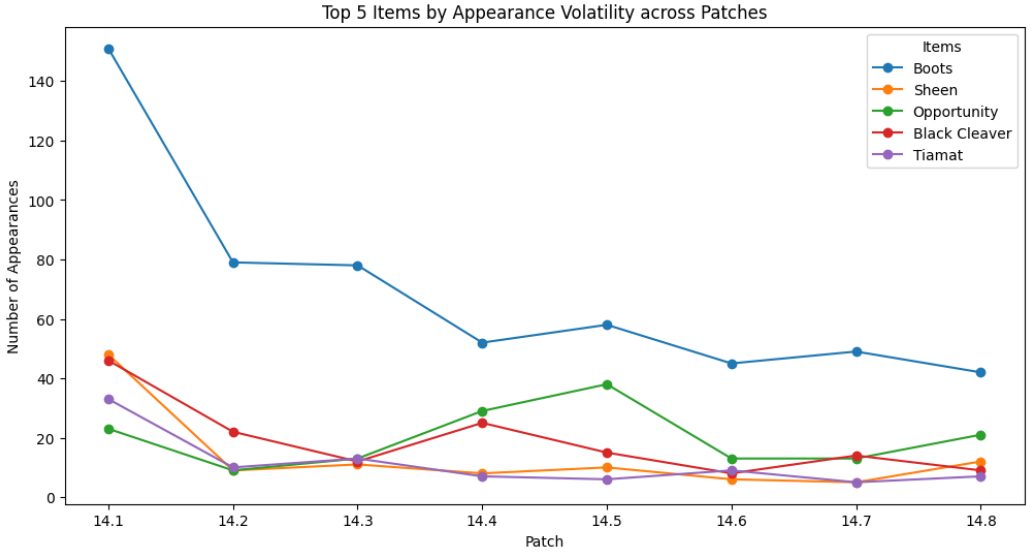


Figure 30: Top 5 Items by Appearance Volatility

Throughout the item analysis from Reddit data, "Boots" emerges as the most frequently mentioned item across all Reddit posts. It ranks among the top five in sentiment rank and appearance volatility, but falls within the bottom five in both score volatility rank and sentiment volatility rank. This prominence could be attributed to the fact that "Boots" is a universally equipped item for all champions in the game, regardless of class, role, or position. The increase in movement speed provided by "Boots" is a valuable stat that benefits any champion, making it a traditionally mandatory purchase to keep up with opponents who also equip the item.

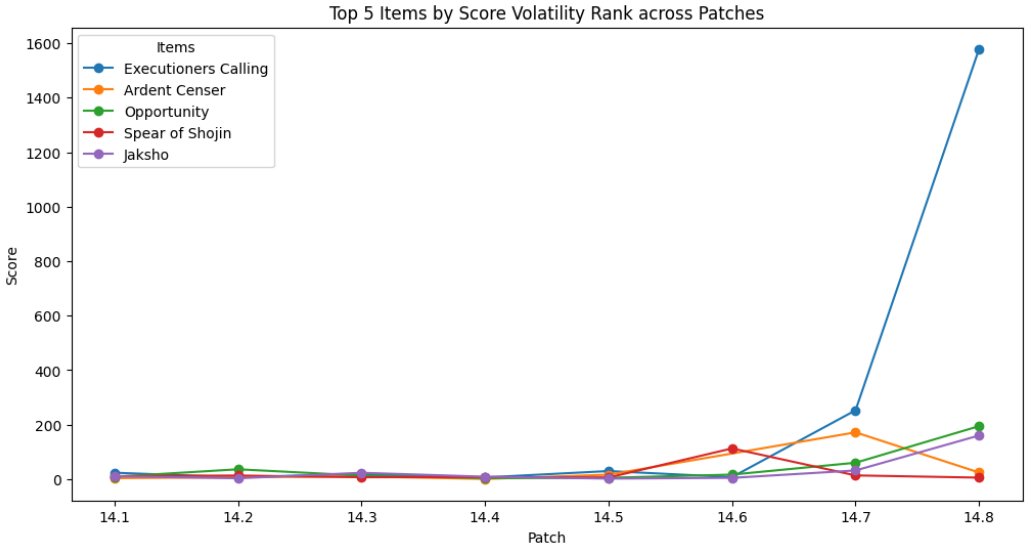


Figure 31: Top 5 Items by Score Volatility Rank

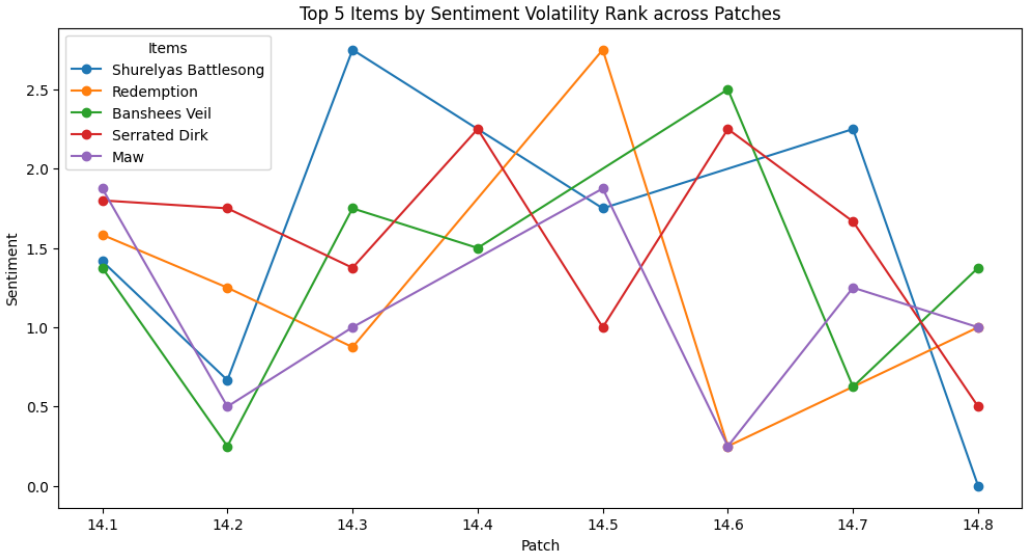


Figure 32: Top 5 Items by Sentiment Volatility Rank

Another item that appears to stand out is "Executioner's Calling". It ranks in the top five items based on score rank, comment rank, and score volatility rank, yet it falls into the bottom five items for sentiment rank. Notably, upvote scores for posts discussing "Executioner's Calling" increase noticeably around patch 14.6, suggesting a strong consensus among Reddit

users about the sentiments expressed. "Executioner's Calling" is an item that applies the so called "Grievous Wounds" effect [74], which reduces healing and health regeneration of enemy champions. The heavy discussion surrounding this item could point to a meta state where champions with high self-healing abilities or support champions with strong healing skills dominate. This may lead to the perception that "Executioner's Calling" is underperforming in addressing these champions and potentially fuel debates about its effectiveness.

Additionally, two items, "Opportunity" and "Eclipse", frequently appear throughout the item analysis. Both are commonly equipped by assassin-class champions. "Opportunity", in particular, is a new item introduced at the start of season 14. The frequent discussions surrounding these items may reflect a heightened community interest in the assassin class.

7.6 Analysis of discussion-dominating Champions

Through a process analogous to what is applied throughout Section 7.5, the League of Legends champion dictionary, introduced in Section 6.4.2, can be utilized to target champions within the Reddit data. Figures 33, 34, 35, and 36 display the top and bottom 5 champions based on number of appearances, score rank, comment rank, and sentiment rank. Additionally, Figures 37, 38, and 39 display the top five champions according to appearance volatility, score volatility rank, and sentiment volatility rank. The bottom five champions based on appearance volatility, score volatility rank, and sentiment volatility rank can be seen in Appendix Q.

In the champion analysis based on Reddit data, "Skarner" emerges as the most frequently mentioned champion, ranking among the top five in appearance volatility but in the bottom five for both score and sentiment volatility. Notably, his appearances increased and sentiment dropped noticeably from patch 14.7 onwards, possibly reflecting the community's reaction to the Skarner VGU, which, as mentioned previously, left the champion in an unbalanced state and sparked heightened discussions. This imbalance seems to have generated dissatisfaction among players, leading to more mentions and a decrease in positive sentiment.

Additionally, champions like "Aatrox", "Kayn", and "Mordekaiser" follow closely in terms of mentions, with "Aatrox" in particular ranking among the top five for appearance and score volatility but in the bottom five for sentiment rank and sentiment volatility rank. Alongside Darius, who ranks

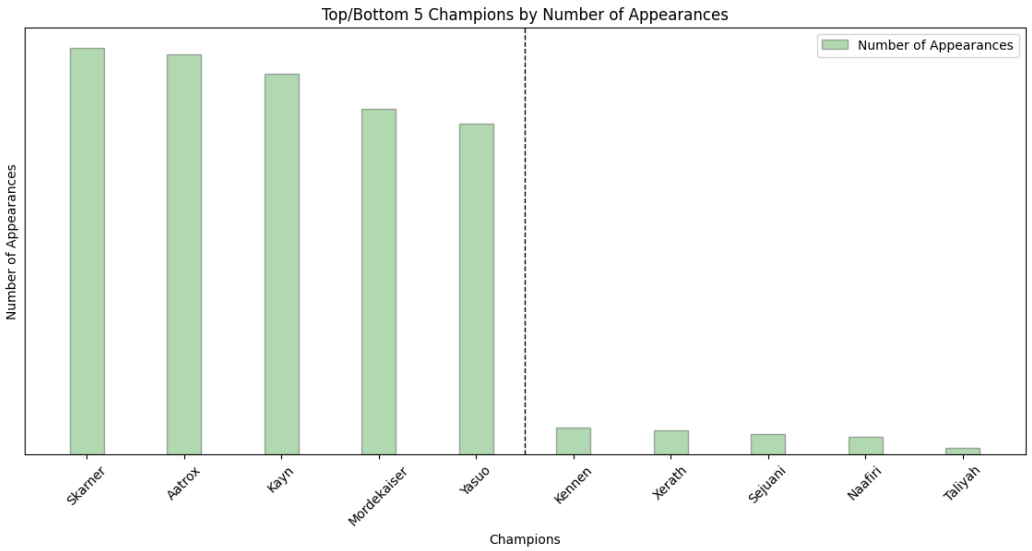


Figure 33: Top/Bottom 5 Champions by Number of Appearance

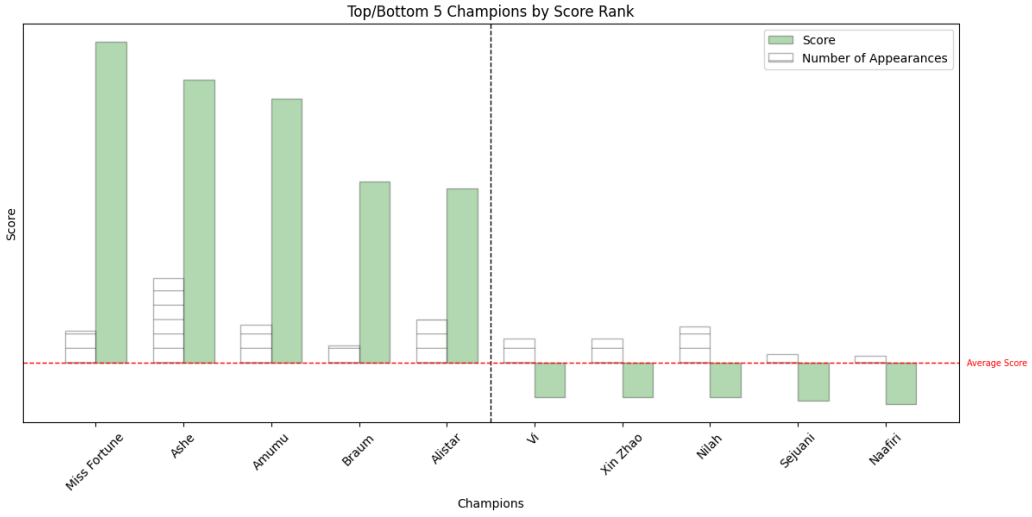


Figure 34: Top/Bottom 5 Champions by Score Rank

high in score volatility, these champions belong to the bruiser/fighter class, and are particularly known for their strong self-healing abilities. The sentiment and extensive discussions around these champions could possibly be tied to the community's dissatisfaction with the above mentioned item "Executioner's Calling", which may be perceived as a weak counter to healing champions. This sentiment is further supported by the inclusion of "Soraka", who ranks among the top five for sentiment volatility. Discussions around

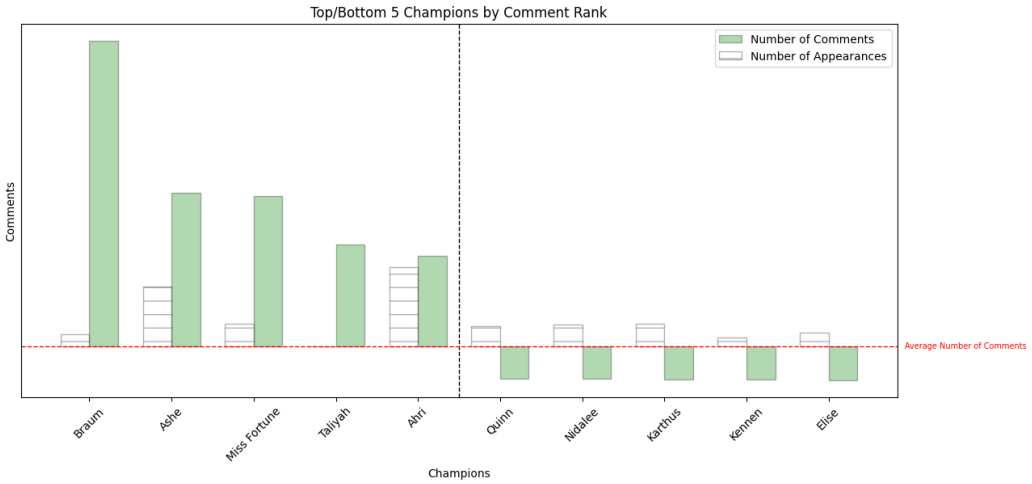


Figure 35: Top/Bottom 5 Champions by Comment Rank

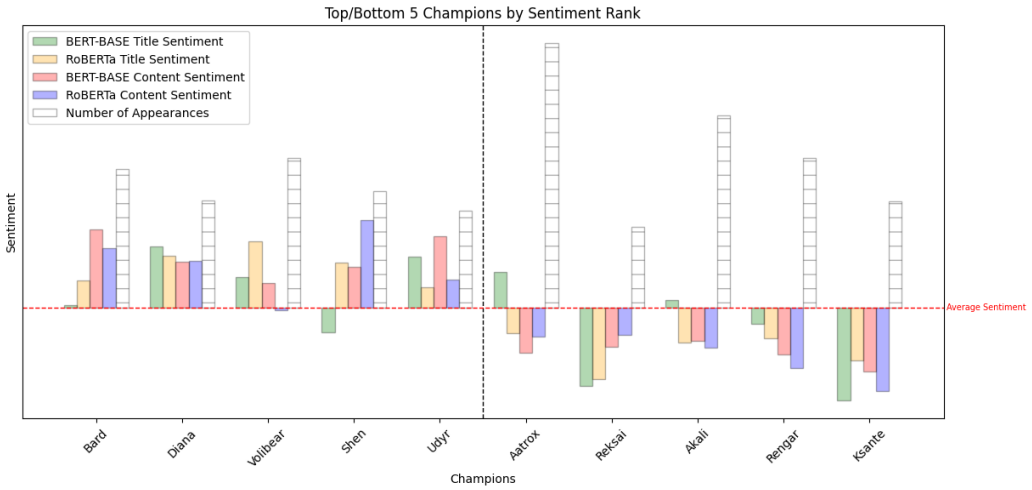


Figure 36: Top/Bottom 5 Champions by Sentiment Rank

this Support-role champion known for her powerful healing abilities, may further indicate community frustration with healing-heavy champions.

A group of assassin champions, including "Naafiri", "Rek'Sai", "Rengar", "Shaco", and "Kayn", also frequently surface in the analysis. "Kayn" is particularly difficult to analyse, as he can evolve into either a bruiser or an assassin, depending on itemization and form selection. His fluctuating win rates across patches 14.4 to 14.8 (as evidenced by scraped data from LoLalytics, METAsrc, U.GG, and visualized in Figure 40), along with the

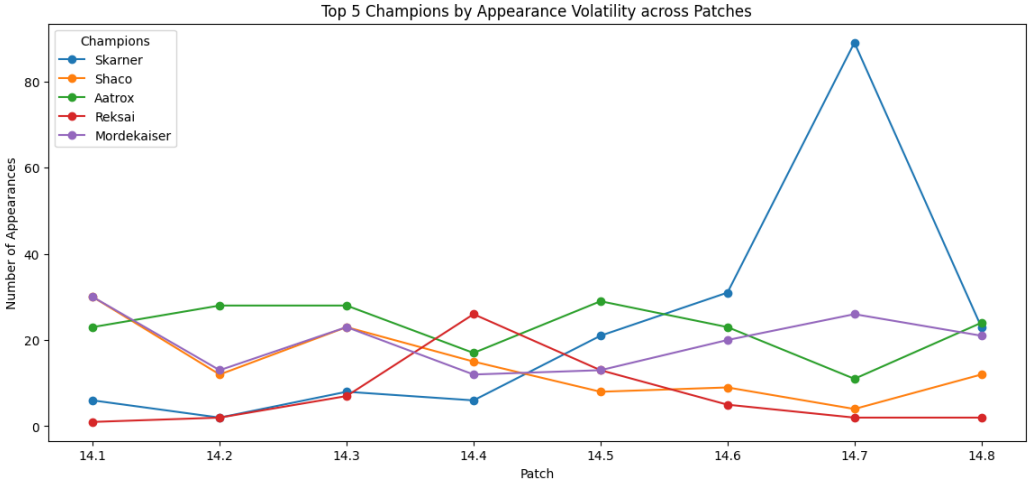


Figure 37: Top 5 Champions by Appearance Volatility

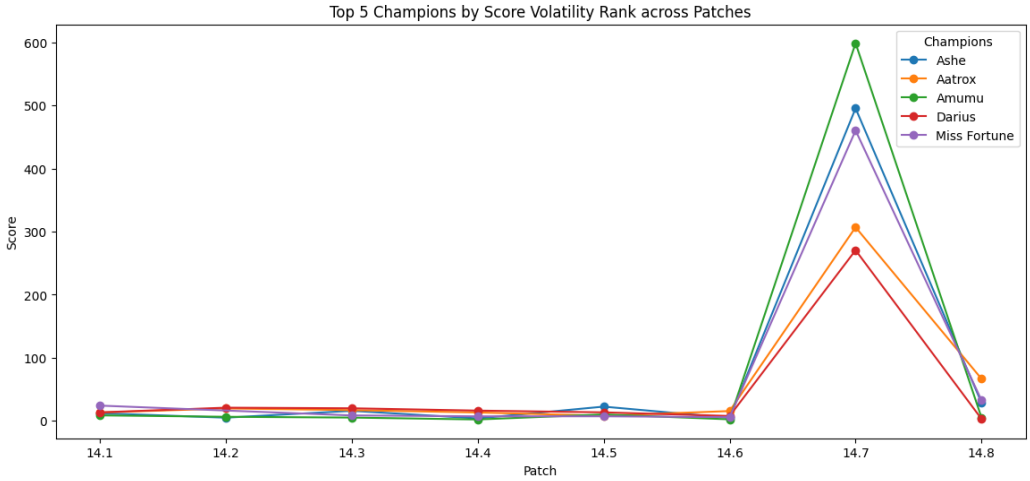


Figure 38: Top 5 Champions by Score Volatility Rank

updates he received in patches 14.5 [84], 14.6 [85], and 14.7 [86], suggest that developers are continuously fine-tuning his balance due to his dual-nature gameplay. The heightened discussions around assassins could also be linked to the ongoing debates around the above mentioned assassin items "Eclipse" and "Opportunity". Specifically, "Naafiri" performed consistently above average in terms of win rates during patches 14.4 to 14.8 (Figure 41), indicating strong performance that may fuel community interest and discussions.

The champion "Yasuo" stands out as the fifth most mentioned cham-

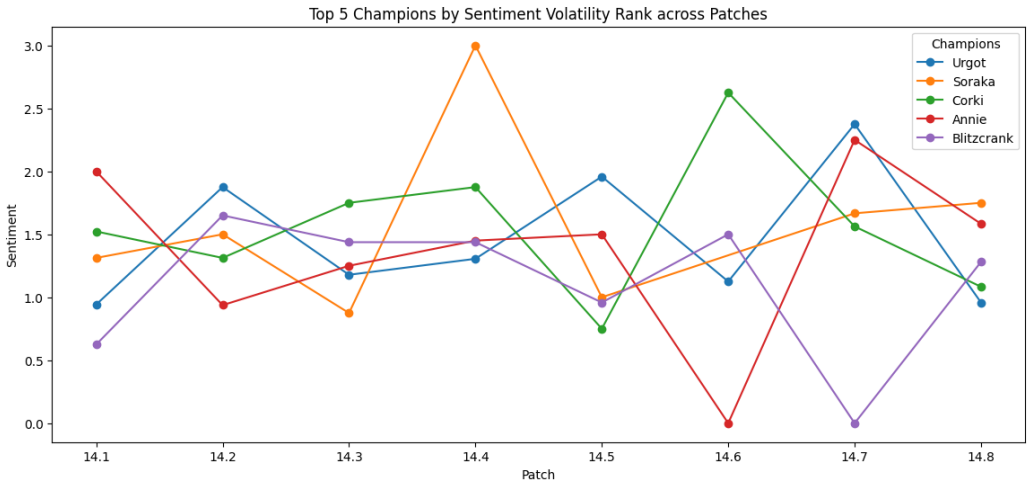


Figure 39: Top 5 Champions by Sentiment Volatility Rank

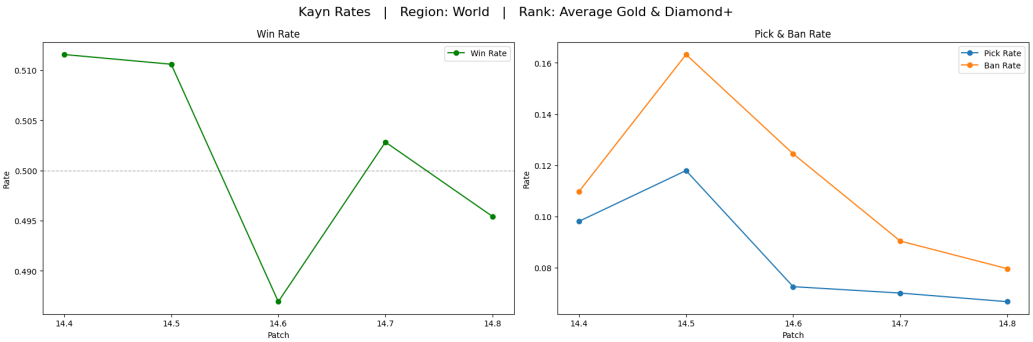


Figure 40: Kayn Win/Pick/Ban Rates across Patches

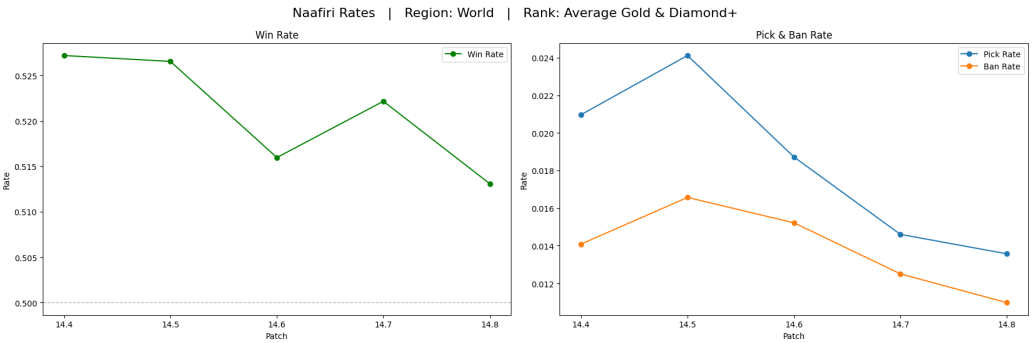


Figure 41: Naafiri Win/Pick/Ban Rates across Patches

pion, ranking in the bottom five for both score and sentiment volatility. The

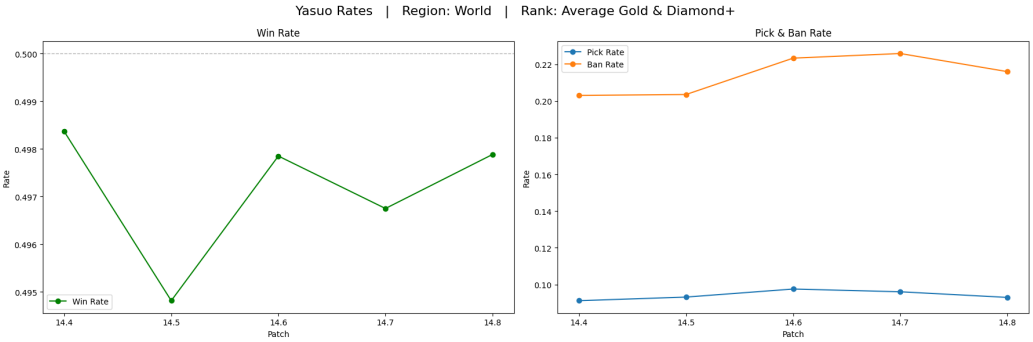


Figure 42: Yasuo Win/Pick/Ban Rates across Patches

champion exhibits high mobility, consistent damage, defensive capabilities, and low health pool. Because of this challenging ability combination, "Yasuo" is infamous for being difficult to master. Despite his below-average win rate, he has a remarkably high ban rate (Figure 42). This might be attributed to a small group of skilled players excelling with "Yasuo", attracting many inexperienced players to try him but ultimately failing due to the champion's complex mechanics. As a result, players may frequently ban "Yasuo" to avoid facing opponents who have mastered him or to prevent inexperienced teammates from underperforming. Additionally, "Yasuo" conventionally equips one or two life-steal items, making him reliant on healing from damage dealt, which aligns with the broader debates about anti-heal items like "Executioner's Calling."

8 Limitations

One technical limitation of this research project stems from the way the Reddit API operates. It allows users to query posts from a subreddit in a way that moves "backwards in time" along the subreddit's timeline, meaning that the most recent posts are returned first. However, there is a quantitative limit on how many posts can be retrieved across multiple queries from a given subreddit. As a result, older posts from popular subreddits like "leagueoflegends" are often inaccessible, and for this subreddit, only posts from the last two patch durations could be collected. This restricts the comprehensiveness of the dataset and skews the analysis toward more recent discussions. Moreover, aggregate champion statistics from third-party sites like LoLalytics, METAsrc, and U.GG are only available for the last four to five patches, in this case specifically from patches 14.4 to 14.8. This further limits the possibility to directly compare Reddit discussions with champion performance

data, particularly for longer-term trends or older patches.

Additionally, the insights generated through this analysis are likely influenced by discrepancies between low and high elo players, as well as the presence of discussions about different game modes apart from the standard 5v5 mode. These variations may distort the conclusions drawn from the data, as the preferences and experiences of different segments of the player base may not align. More broadly, it's important to recognize that this approach is intended to illustrate a complementary toolset for game developers to adapt updates based on community feedback. However, determining which parts of the player base to prioritize remains a challenging question, given the competitive nature of the game and its varied audience. Furthermore, the results from this analysis should only be interpreted by individuals with sufficient domain knowledge about League of Legends and its balance state. Lastly, because this analysis is based on anonymous, user-generated content from Reddit, the veracity of the underlying data is inherently volatile, which adds another layer of complexity to the accuracy and reliability of the findings.

9 Conclusion and Further Research

The data analysis presented in this study highlights the important role that community sentiment and player feedback can play in informing the development and updating processes for games and software products. The case study of League of Legends, demonstrates how textual data from social media platforms (specifically Reddit) can be mined and analyzed to provide insights into user opinions, preferences, and issues. This type of analysis highlights key areas of concern and enables developers to respond more swiftly and effectively to community demands. Specifically, this study aims to answer the following central research question: "How can text mining and sentiment analysis of textual Reddit data be used to inform data-driven game updates for League of Legends, with the aim of enhancing player satisfaction and thereby driving economic success?" It is shown, that by mining and analyzing Reddit data, developers can identify key trends in community sentiment and specific issues players are facing. These insights allow for more targeted updates and improvements that not only address player concerns but also foster long-term user engagement, which is critical for the economic success of the League of Legends.

While the case of League of Legends provides a focused lens through which this concept approach is explored, the findings and methodologies employed

are applicable to a broader spectrum of software products, particularly in the highly competitive world of video game development. Extrapolating these findings beyond League of Legends suggests a number of important lessons for the video game industry as a whole, as well as for other sectors within the software market that rely on community engagement, frequent updates, and a balance between user satisfaction and business objectives.

One of the key takeaways from this analysis is that social media platforms, particularly those like Reddit, offer a large corpus of user-generated content that can provide developers with unfiltered feedback. In the modern video game and software market, players and users no longer engage with products in a vacuum [22]. They interact with both the product and the surrounding community, and they frequently share their thoughts, frustrations, and suggestions online. Reddit, with its unique structure of subreddits dedicated to specific games or software products, offers a valuable source of community feedback that developers can use to inform product updates and future development decisions. The ability to analyze this data in a structured way, through techniques like text mining and sentiment analysis, offers software developers an opportunity to engage with their user base in a more meaningful and data-driven manner. Instead of relying solely on small-scale feedback loops such as surveys, beta tests, or focus groups, developers can use social media to tap into a much larger and more diverse set of voices. This is particularly relevant in the case of "software as a service", where the user base is constantly evolving, and the needs and desires of the community can shift rapidly in response to new content or updates. Moreover, this type of analysis allows developers to identify not just broad trends in community sentiment, but also more granular insights. As seen in the case of League of Legends, different subreddits often host distinct player communities, each with their own preferences, playstyles, and expectations. By analyzing discussions across these different forums, developers can gain a more nuanced understanding of how different segments of their player base respond to changes in the game. This insight can be useful when it comes to making decisions about future updates, as it allows developers to tailor their responses to specific sub-communities and ensure that a broader range of users is satisfied. For instance, in the analysis of League of Legends patches, players' negative reactions to the Skarner VGU in patch 14.7 become evident through the noticeable rise in upvote scores and the simultaneous decline in sentiment. This type of feedback can act as a red flag for developers, indicating that a large-scale update has not been well-received by the community and requires further adjustments or balancing. Similarly, the prominence of discussions around certain items and champion classes in the game point

to areas where the community feels dissatisfied or frustrated, offering clear opportunities for targeted improvements in future patches.

The ability to quickly and effectively respond to player feedback is not just a matter of community engagement. It also has significant economic implications for game developers and the broader software industry. In the modern marketplace, where players have access to a large variety of games and software products, maintaining user engagement is critical for driving revenue [56]. A game or software product that fails to keep its user base satisfied runs the risk of losing users to competitors, resulting in a decline in both active users and revenue streams. Conversely, services that are able to successfully adapt to their community's needs and preferences can build long-term loyalty, which in turn drives sustained revenue growth [7]. The rise of "software as a service" has shifted the focus from one-time purchases to ongoing revenue models, such as microtransactions, in-software purchases, subscriptions, and expansions. In these models, user retention is key to profitability. A user who remains engaged with a game or software product over the long term is more likely to make in-software purchases [40][33][19]. For this reason, developers who are able to leverage data-driven insights to fine-tune their updates and keep users interested are better positioned for financial success. Furthermore, a more responsive and data-driven approach to updates can reduce the risk of large-scale user dissatisfaction following a major update. As seen in the case of the Skarner VGU, player sentiment took a visible downturn following the release of a controversial update. If this type of feedback goes unnoticed or unaddressed over a longer period, it could lead to widespread frustration and, ultimately, a loss of player retention. However, by paying close attention to community sentiment and responding accordingly, developers can mitigate the risk of loosing their player base, which in turn protects the game's long-term revenue potential. A company that fails to respond to its users risks losing market share to competitors who are more attuned to their user base. On the other hand, a company that actively incorporates user feedback into its development process is more likely to foster loyalty and generate profit.

The methodological approach presented in this thesis is not limited to video games; it has broad applicability across the software industry. For any software product that relies on ongoing updates, whether it's a mobile app, an online software platform, or a social networking site, community feedback is a vital component of product development. While the focus of this case study has been on a specific game within the video game industry, the methodologies employed here have broader relevance for the software market

as a whole. For example, a mobile app developer could use sentiment analysis to track how users are reacting to a new version of their app following an update. If negative sentiment begins to rise shortly after the release, this could be a signal that the update has introduced bugs or features that users find frustrating. By identifying these trends early, the developer can address the issue before it results in widespread user churn.

Looking forward, it is likely that the use of data-driven feedback will become an increasingly important component of game and software development. As machine learning and natural language processing techniques continue to evolve, developers will have access to more powerful tools for analyzing user generated content and feedback. In the future, we may see more sophisticated forms of sentiment analysis that can track how different updates affect various segments of a user base [11]. Moreover, the growing importance of "software as a service" models means that companies will need to become more responsive to their users. As competition in these markets intensifies, the ability to rapidly and accurately respond to community feedback will be a key differentiator between successful and unsuccessful products. In conclusion, the findings from this case study on League of Legends provide valuable insights into the broader potential of data-driven game and software development. By leveraging social media platforms like Reddit, developers can gain real-time insights into community sentiment, identify areas of concern, and respond more effectively to player or user feedback. This, in turn, has significant economic implications, as products that are able to keep their users engaged over the long term are more likely to generate sustained revenue. The methodologies and insights learned from this study have the potential to be applied across a wide range of software products, making data-driven feedback an essential tool for developers.

This research approach also offers a foundation for future research into how player-generated content can be used to inform game updates. One major area for expansion involves improving the granularity of sentiment analysis. Currently, models like BERT base and RoBERTa provide a useful but limited view of community sentiment by categorizing it into broad trends. In future research, developers could refine these models or integrate new ones that capture more nuanced emotional reactions, such as frustration, excitement, or confusion, rather than simply positive or negative sentiments. This could allow developers to pinpoint not only when players are dissatisfied, but also why.

Future research could also directly expand on the present approach by in-

corporating more sophisticated methods to analyze Reddit comments, such as applying TF-IDF term clustering specifically to the comment data. This would allow for a more granular understanding of the discussions surrounding items, champions, and patches, providing developers with even deeper insights into how various updates are perceived. Moreover, moving beyond TF-IDF, implementing Latent Dirichlet Allocation (LDA) topic modeling could help identify more specific topics or themes within the Reddit community. By automatically clustering discussions into coherent topics, LDA could reveal patterns in player feedback, such as common concerns or suggestions related to certain game mechanics or updates.

Additionally, future research could focus on filtering and analyzing Reddit discussions based on different game modes, such as the standard 5v5 mode, ARAM, or URF. Different game modes may generate different types of feedback, and separating these modes in the analysis could allow developers to tailor their updates to specific gameplay experiences. For example, certain items or champions may perform differently across modes, and filtering the analysis would highlight mode-specific issues that might otherwise be overlooked. Beyond champion performance, utilizing Riot Games' Data Dragon tool to investigate item changes in detail could offer further valuable insights. This would complement the current analysis focused on champion changes by capturing how item updates impact gameplay and community reactions.

By integrating these advanced analytical methods and expanding the scope of the data being examined, more actionable insights for developers could be provided. This extended approach would help identify specific issues with champions, items, and game modes more precisely, ultimately allowing for more responsive and targeted game updates that align with community feedback.

References

[1] Jesús C Aguerri, Mario Santisteban, and Fernando Miró-Llinares. The enemy hates best? toxicity in league of legends and its content moderation implications. *European Journal on Criminal Policy and Research*, 29(3):437–456, 2023.

[2] Kati Alha, Elina Koskinen, Janne Paavilainen, Juho Hamari, and Jani Kinnunen. Free-to-play games: Professionals' perspective. *Proceedings of DiGRA Nordic 2014*, 2014.

[3] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saied Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919*, 2017.

[4] Katie Elson Anderson. Ask me anything: what is reddit? *Library Hi Tech News*, 32(5):8–11, 2015.

[5] Chee Siang Ang, Panayiotis Zaphiris, and Stephanie Wilson. Computer games and sociocultural play: An activity theoretical perspective. *Games and Culture*, 5(4):354–380, 2010.

[6] Eva Ascarza, Oded Netzer, and Julian Runge. Personalized game design for improved user retention and monetization in freemium mobile games. *Available at SSRN 4653319*, 2023.

[7] Janarthanan Balakrishnan and Mark D Griffiths. Loyalty towards online games, gaming addiction, and purchase intention towards online mobile in-game features. *Computers in Human Behavior*, 87:238–246, 2018.

[8] Muneera Bano, Didar Zowghi, and Francesca da Rimini. User satisfaction and system success: an empirical exploration of user involvement in software development. *Empirical Software Engineering*, 22:2339–2372, 2017.

[9] C. Bateman. The anarchy of paidia. https://onlyagame.typepad.com/only_a_game/2005/12/the_anarchy_of__1.html, 2005. Accessed: 2024-08-01.

[10] Emily Joy Bembeneck. Game, narrative and storyworld in league of legends. *The Play Versus Story Divide in Game Studies: Critical Essays*, page 51, 2015.

[11] Jeremy Blackburn and Haewoon Kwak. Stfu noob! predicting crowdsourced decisions on toxic behavior in online games. In *Proceedings of the 23rd international conference on World wide web*, pages 877–888, 2014.

[12] Bryce Boe. Praw 7.7.1 documentation. https://praw.readthedocs.io/en/stable/, 2023. Accessed: 2024-08-31.

[13] cardiffnlp. Tweetnlp. https://github.com/cardiffnlp/tweetnlp, 2022. Accessed: 2024-09-02.

[14] cardiffnlp. Twitter-roberta-base for sentiment analysis - updated (2022). https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest, 2022. Accessed: 2024-05-31.

[15] Sua Choi. The design of 'league of legends' decision support system for inexperienced players. B.S. thesis, University of Twente, 2022.

[16] Ismail Civelek, Yipeng Liu, and Sean R Marston. Design of free-to-play mobile games for the competitive marketplace. *International Journal of Electronic Commerce*, 22(2):258–288, 2018.

[17] Mark Claypool, Jonathan Decelle, Gabriel Hall, and Lindsay O'Donnell. Surrender at 20? matchmaking in league of legends. In *2015 IEEE Games Entertainment Media Conference (GEM)*, pages 1–4. IEEE, 2015.

[18] Mark Claypool, Artian Kica, Andrew La Manna, Lindsay O'Donnell, and Tom Paolillo. On the impact of software patching on gameplay for the league of legends computer game. *The Computer Games Journal*, 6:33–61, 2017.

[19] Jack Cleghorn and Mark D Griffiths. Why do gamers buy "virtual assets"? an insight in to the psychology behind purchase behaviour. *Digital Education Review*, 27:85–104, 2015.

[20] PyTorch Contributors. Pytorch documentation. https://pytorch.org/docs/2.3/, 2023. Accessed: 2024-09-02.

[21] Tiffany D Do, Seong Ioi Wang, Dylan S Yu, Matthew G McMillian, and Ryan P McMahan. Using machine learning to predict game outcomes based on player-champion experience in league of legends. In *Proceedings of the 16th International Conference on the Foundations of Digital Games*, pages 1–5, 2021.

[22] Scott Donaldson. Mechanics and metagame: Exploring binary expertise in league of legends. *Games and Culture*, 12(5):426–444, 2017.

[23] Joris Dormans. Integrating emergence and progression. In *Proceedings of DiGRA 2011 Conference: Think design play*, 2011.

[24] Simon Ferrari. From generative to conventional play: Moba and league of legends. In *Proceedings of DiGRA 2013 Conference*, 2013.

[25] Casey Fiesler, Jialun Jiang, Joshua McCann, Kyle Frye, and Jed Brubaker. Reddit rules! characterizing an ecosystem of governance. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 12, 2018.

[26] Irta Fitriana and Misbakhul Munir Rois. Analysis of metaphors applied in mobile legends game. *Diglossia: Jurnal Kajian Ilmiah Kebahasaan dan Kesusastraan*, 15(1):118–132, 2023.

[27] Chek Yang Foo and Elina MI Koivisto. Defining grief play in mmorpgs: player and developer perceptions. In *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 245–250, 2004.

[28] Nick Galov. What's the most popular moba-league of legends player count. *Web Tribunal*, 2023.

[29] Riot Games. Data dragon. https://riot-api-libraries.readthedocs.io/en/latest/ddragon.html, 2019. Accessed: 2024-09-02.

[30] Riot Games. https://www.riotgames.com/en, 2024. Accessed: 2024-07-29.

[31] Riot Games. Patch notes. https://www.leagueoflegends.com/en-us/news/tags/patch-notes/, 2024. Accessed: 2024-08-08.

[32] Bachan Ghimire, Ze Shi Li, and Daniela Damian. Understanding user feedback in software ecosystems: A study on challenges and mitigation strategies. In *International Conference on Software Business*, pages 132–147. Springer, 2023.

[33] Mark D Griffiths. The role of context in online gaming excess and addiction: Some case study evidence. *International journal of mental health and addiction*, 8:119–125, 2010.

[34] gutz505. Lol-reddit-analysis. https://github.com/gutz505/LoL-Reddit-Analysis, 2024. Accessed: 2024-09-18.

[35] Juho Hamari, Kati Alha, Simo Järvelä, J Matias Kivikangas, Jonna Koivisto, and Janne Paavilainen. Why do players buy in-game content? an empirical study on concrete purchase motivations. *Computers in Human Behavior*, 68:538–546, 2017.

[36] Juho Hamari and Vili Lehdonvirta. Game design as marketing: How game mechanics create demand for virtual goods. *International Journal of Business Science & Applied Management*, 5(1):14–29, 2010.

[37] Yuzi He, Christopher Tran, Julie Jiang, Keith Burghardt, Emilio Ferrara, Elena Zheleva, and Kristina Lerman. Heterogeneous effects of software patches in a multiplayer online battle arena game. In *Proceedings of the 16th International Conference on the Foundations of Digital Games*, pages 1–9, 2021.

[38] Brian M Hook. Statistics in league of legends: Analyzing runes for last-hitting. Augustana College, Rock Island Illinois, Spring 5-17-2016, 2016.

[39] Kuo-Lun Hsiao and Chia-Chen Chen. What drives in-app purchase intention for mobile games? an examination of perceived values and loyalty. *Electronic commerce research and applications*, 16:18–29, 2016.

[40] Emil R Kaburuan, Chien-Hsu Chen, and Tay-Sheng Jeng. Identifying users' behavior purchasing virtual items. *Researchers World - Journal of Arts Science & Commerce*, E-ISSN 2229-4686, 2009.

[41] Burak KARACA, Zarife Pancar, Buğra Anarefe, and Mehmet Şenoğlu. A research on club performances in esports: The example of the league of legends 2022 world cup. *Research Square*, https://doi.org/10.21203/rs.3.rs-3879908/v1, 2024.

[42] Liew Ching Kho, Mohd Shareduwan Mohd Kasihmuddin, Mohd Mansor, Saratha Sathasivam, et al. Logic mining in league of legends. *Pertanika Journal of Science & Technology*, 28(1), 2020.

[43] Hee-Woong Kim, Hock Chuan Chan, and Atreyi Kankanhalli. What motivates people to purchase digital items on virtual community websites? the desire for online self-presentation. *Information systems research*, 23(4):1232–1245, 2012.

[44] Athanasios Kokkinakis, Peter York, Moni Sagarika Patra, Justus Robertson, Ben Kirman, Alistair Coates, Alan Pedrassoli Pedrassoli Chitayat, Simon Demediuk, Anders Drachen, Jonathan Hook, et al. Metagaming and metagames in esports. *International Journal of Esports*, 1(1), 2021.

[45] Bastian Kordyaka and Sidney Hribersek. Crafting identity in league of legends-purchases as a tool to achieve desired impressions. *Proceedings of the 52nd Hawaii International Conference on System Sciences*, URI: https://hdl.handle.net/10125/59591, 2019.

[46] Yubo Kou and Xinning Gui. Playing with strangers: understanding temporary teams in league of legends. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, pages 161–169, 2014.

[47] Yubo Kou, Xinning Gui, and Yong Ming Kow. Ranking practices and distinction in league of legends. In *Proceedings of the 2016 annual symposium on computer-human interaction in play*, pages 4–9, 2016.

[48] Yubo Kou, Yao Li, Xinning Gui, and Eli Suzuki-Gill. Playing with streakiness in online games: how players perceive and react to winning and losing streaks in league of legends. In *Proceedings of the 2018 CHI conference on human factors in computing systems*, pages 1–14, 2018.

[49] Yubo Kou and Bonnie A Nardi. Governance in league of legends: A hybrid system. *FDG*, 7:1, 2014.

[50] Kriswanda Krishnapatria. From unsung hero to savage hero: Shaping characters' roles in the adaptation between literary works and mobile legends: Bang bang! moba. *Journal of English Language, Literature, and Teaching*, 3:1, 2019.

[51] Vineet Kumar. Making "freemium" work. *Harvard business review*, 92(5):27–29, 2014.

[52] Choong-Soo Lee and Ivan Ramler. Investigating the impact of game features on champion usage in league of legends. In *FDG*, 2015.

[53] Choong-Soo Lee and Ivan Ramler. Identifying and evaluating successful non-meta strategies in league of legends. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, pages 1–6, 2017.

[54] Vili Lehdonvirta. Virtual item sales as a revenue model: Identifying attributes that drive purchase decisions. *Electronic commerce research*, 9:97–113, 2009.

[55] Holin Lin and Chuen-Tsai Sun. Cash trade within the magic circle: Free-to-play game challenges and massively multiplayer online game player responses. In *Proceedings of DiGRA 2007 Conference: Situated Play*, 2007.

[56] Hsin-Hui Lin and Yi-Shun Wang. An examination of the determinants of customer loyalty in mobile commerce contexts. *Information & management*, 43(3):271–282, 2006.

[57] Yile Liu, Yitao Ma, and Tianhui Wang. The spread of league of legends. In *2021 International Conference on Public Art and Human Development (ICPAHD 2021)*, pages 133–137. Atlantis Press, 2022.

[58] Lolalyitics. https://lolalytics.com/, 2024. Accessed: 2024-08-04.

[59] Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-Collados. Timelms: Diachronic language models from twitter. *arXiv preprint arXiv:2202.03829*, 2022.

[60] Heather L Lynch. *Composing a gamer: A case study of one gamer's experience of symbiotic flow*. Georgia State University, 2013.

[61] Ben Marder, David Gattig, Emily Collins, Leyland Pitt, Jan Kietzmann, and Antonia Erz. The avatar's new clothes: Understanding why players purchase non-functional items in free-to-play games. *Computers in Human Behavior*, 91:72–83, 2019.

[62] George Margetis, Konstantinos C Apostolakis, Stavroula Ntoa, Ioannis Chatzakis, Eirini Sykianaki, Ioannis Markopoulos, and Constantine Stephanidis. Visual summarisations for computer-assisted live color casting and direction in league of legends. In *International Conference on Human-Computer Interaction*, pages 133–153. Springer, 2023.

[63] Jose Luis Marín de la Iglesia and Jose Emilio Labra Gayo. Doing business by selling free services. In *Web 2.0: The Business Model*, pages 1–14. Springer, 2008.

[64] Jordan Marney. League of legends map explained. https://www.esports.net/news/lol/league-of-legends-map-explained/, 2024. Accessed: 2024-09-05.

[65] Philip Z Maymin. Smart kills and worthless deaths: esports analytics for league of legends. *Journal of Quantitative Analysis in Sports*, 17(1):11–27, 2021.

[66] Alexey N Medvedev, Renaud Lambiotte, and Jean-Charles Delvenne. The anatomy of reddit: An overview of academic research. *Dynamics on and of Complex Networks III: Machine Learning and Statistical Physics Approaches 10*, pages 183–204, 2019.

[67] Heather L Mello. Invoking the avatar: Gaming skills as cultural and out-of-game capital. *Gaming as culture: Essays on reality, identity and experience in fantasy games*, pages 175–95, 2006.

[68] METAsrc. https://www.metasrc.com/, 2024. Accessed: 2024-08-04.

[69] Marcal Mora-Cantallops and Miguel-Ángel Sicilia. Team efficiency and network structure: The case of professional league of legends. *Social Networks*, 58:105–115, 2019.

[70] Bryn Neuenschwander. Playing by the rules: Instruction and acculturation in role-playing games. *E-Learning and Digital Media*, 5(2):189–198, 2008.

[71] nlptown. bert-base-multilingual-uncased-sentiment. https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment, 2020. Accessed: 2024-05-31.

[72] Miho Nojima. Pricing models and motivations for mmo play. In *Proceedings of DiGRA 2007 Conference: Situated Play*, 2007.

[73] NumFOCUS. pandas documentation. https://pandas.pydata.org/docs/index.html, 2024. Accessed: 2024-08-31.

[74] League of Legends Wiki. Grievous wounds. https://leagueoflegends.fandom.com/wiki/Grievous_Wounds, 2024. Accessed: 2024-09-04.

[75] League of Legends Wiki. Item (league of legends). https://leagueoflegends.fandom.com/wiki/Item_(League_of_Legends), 2024. Accessed: 2024-08-04.

[76] League of Legends Wiki. List of champions. https://leagueoflegends.fandom.com/wiki/List_of_champions, 2024. Accessed: 2024-07-29.

[77] League of Legends Wiki. Summoner's code. https://leagueoflegends.fandom.com/wiki/Summoner%27s_Code, 2024. Accessed: 2024-08-02.

[78] Cindy Poremba. Player as author: Digital games and agency. 2003.

[79] Nicholas Proferes, Naiyan Jones, Sarah Gilbert, Casey Fiesler, and Michael Zimmer. Studying reddit: A systematic overview of disciplines, approaches, methods, and ethics. *Social Media+ Society*, 7(2):20563051211019004, 2021.

[80] Risto Rajala, Matti Rossi, Virpi Kristiina Tuunainen, and Janne Vihinen. Revenue logics of mobile entertainment software–observations from companies producing mobile games. *Journal of Theoretical and Applied Electronic Commerce Research*, 2(2):34–47, 2007.

[81] Jana Rambusch, Peter Jakobsson, and Daniel Pargman. Exploring e-sports: A case study of gameplay in counter-strike. In *Proceedings of DiGRA 2007 Conference: Situated Play*, 2007.

[82] reddit_irl. Reddit recap 2023. https://new.reddit.com/r/recap/comments/18c4kvr/keeping_it_dialed_in_2023_redditors_sought_honest/, 2023. Accessed: 2024-08-10.

[83] Leonard Richardson. Beautiful soup documentation. https://www.crummy.com/software/BeautifulSoup/bs4/doc/, 2023. Accessed: 2024-09-02.

[84] Riot Riru. Patch 14.5 notes. https://www.leagueoflegends.com/en-gb/news/game-updates/patch-14-5-notes/, 2024. Accessed: 2024-09-04.

[85] Riot Riru. Patch 14.6 notes. https://www.leagueoflegends.com/en-gb/news/game-updates/patch-14-6-notes/, 2024. Accessed: 2024-09-04.

[86] Riot Sakaar. Patch 14.7 notes. https://www.leagueoflegends.com/en-us/news/game-updates/patch-14-7-notes/, 2024. Accessed: 2024-09-03.

[87] Scarizard. Patch 5.16 notes. https://web.archive.org/web/20160122202602/http://na.leagueoflegends.com/en/news/game-updates/patch/patch-516-notes, 2015. Accessed: 2024-08-05.

[88] Fahim Shahriar. Riot talks about social media feedback regarding league of legends. https://gameriv.com/social-media-feedback-league-of-legends/, 2024. Accessed: 2024-09-09.

[89] Carl Shapiro and Hal R Varian. *Information rules: A strategic guide to the network economy*. Harvard Business Press, 1999.

[90] Neeraj Anand Sharma, ABM Shawkat Ali, and Muhammad Ashad Kabir. A review of sentiment analysis: tasks, applications, and deep learning techniques. *International Journal of Data Science and Analytics*, pages 1–38, 2024.

[91] Sugam Sharma. Evolution of as-a-service era in cloud. *arXiv preprint arXiv:1507.00939*, 2015.

[92] Cuihua Shen and Dmitri Williams. Unpacking time online: Connecting internet and massively multiplayer online game use with psychosocial well-being. *Communication Research*, 38(1):123–149, 2011.

[93] Yusaku Shibata, Hirofumi Kurihara, and Shinzo Takatsu. Towards a concept of meta-game: Some applied results. In *Global Interdependence: Simulation and Gaming Perspectives Proceedings of the 22nd International Conference of the International Simulation and Gaming Association (ISAGA) Kyoto, Japan: 15–19 July 1991*, pages 49–56. Springer, 1992.

[94] Mirna Paula Silva, Victor do Nascimento Silva, and Luiz Chaimowicz. Dynamic difficulty adjustment on moba games. *Entertainment Computing*, 18:103–123, 2017.

[95] Skittle Sniper. Player reporting guide and faq. https://support-leagueoflegends.riotgames.com/hc/en-us/articles/201752884-Player-Reporting-Guide-and-FAQ, 2024. Accessed: 2024-08-02.

[96] Tina Lynn Taylor. *Raising the stakes: E-sports and the professionalization of computer gaming*. Mit Press, 2012.

[97] Katie Salen Tekinbas and Eric Zimmerman. *Rules of play: Game design fundamentals*. MIT press, 2003.

[98] Katie Salen Tekinbas and Eric Zimmerman. *The game design reader: A rules of play anthology*. MIT press, 2005.

[99] Sunny Thaicharoen, Jeremy Gow, and Anders Drachen. An ecosystem framework for the meta in esport games. *Journal of Electronic Gaming and Esports*, 1(1), 2023.

[100] Jarvis the NPC. League of legends: The skarner dilemma - is he overpowered or just fun? https://www.zleague.gg/theportal/league-of-legends-the-skarner-dilemma-is-he-overpowered-or-just-fun/, 2024. Accessed: 2024-09-04.

[101] Twitch. https://www.twitch.tv/, 2024. Accessed: 2024-07-29.

[102] U.GG. https://u.gg/, 2024. Accessed: 2024-08-04.

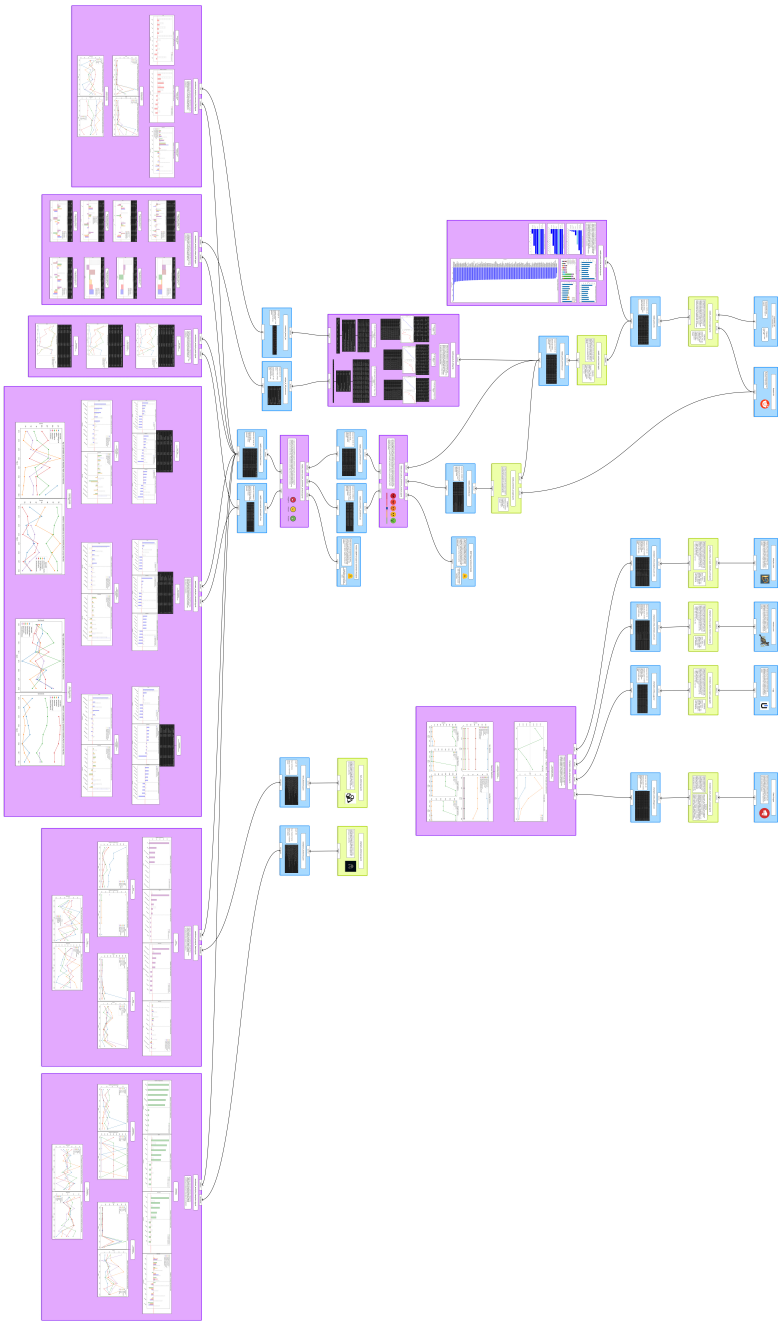
[103] Zonghan Wang. Taking the moba (multiplayer online battle arena) game" league of legends" as a study case. *Journal of Innovation and Development*, 5(1):55–58, 2023.

[104] Steven Daniel Webb and Sieteng Soh. Cheating in networked computer games: a review. In *Proceedings of the 2nd international conference on Digital interactive media in entertainment and arts*, pages 105–112, 2007.

[105] Jae Mee Yoo. Perceived value of game items and purchase intention. *Indian journal of science and technology*, 8(19):1–7, 2015.

Appendix

A Detailed Methodology Overview Diagram



B List of Subreddits

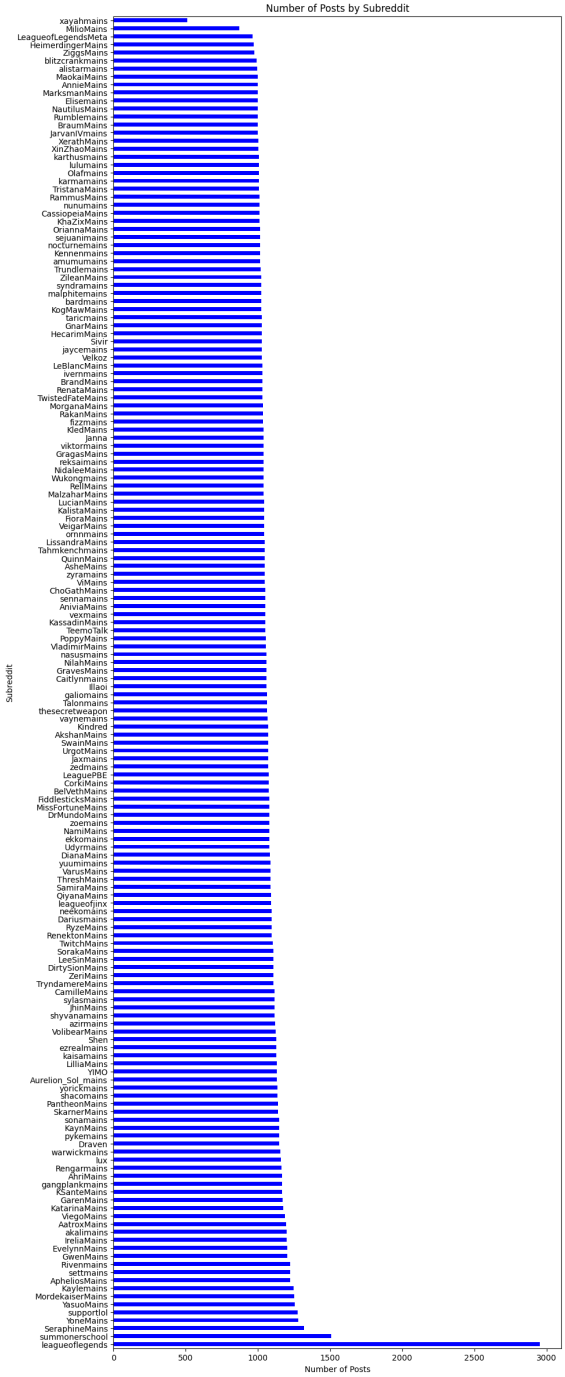
Subreddit Name	Meta Type	URL
summonerschool	General Meta	https://www.reddit.com/r/summonerschool/
leagueoflegends	General Meta	https://www.reddit.com/r/leagueoflegends/
LeagueofLegendsMeta	General Meta	https://www.reddit.com/r/LeagueofLegendsMeta/
LeaguePBE	General Meta (! meta to be implemented !)	https://www.reddit.com/r/LeaguePBE/
MarksmanMains	Champion Class Meta	https://www.reddit.com/r/MarksmanMains/
supportlol	Role Meta	https://www.reddit.com/r/supportlol/
AatroxMains	Champion Meta (Aatrox)	https://www.reddit.com/r/AatroxMains/
AhriMains	Champion Meta (Ahri)	https://www.reddit.com/r/AhriMains/
akalimains	Champion Meta (Akali)	https://www.reddit.com/r/akalimains/
AkshanMains	Champion Meta (Akshan)	https://www.reddit.com/r/AkshanMains/
alistarmains	Champion Meta (Alistar)	https://www.reddit.com/r/alistarmains/
amumumains	Champion Meta (Amumu)	https://www.reddit.com/r/amumumains/
AniviaMains	Champion Meta (Anivia)	https://www.reddit.com/r/AniviaMains/
AnnieMains	Champion Meta (Annie)	https://www.reddit.com/r/AnnieMains/
ApheliosMains	Champion Meta (Aphelios)	https://www.reddit.com/r/ApheliosMains/
AsheMains	Champion Meta (Ashe)	https://www.reddit.com/r/AsheMains/
Aurelion_Sol_mains	Champion Meta (Aurelion Sol)	https://www.reddit.com/r/Aurelion_Sol_mains/
azirmains	Champion Meta (Azir)	https://www.reddit.com/r/azirmains/
bardmains	Champion Meta (Bard)	https://www.reddit.com/r/bardmains/
BelVethMains	Champion Meta (Bel'Veth)	https://www.reddit.com/r/BelVethMains/
blitzcrankmains	Champion Meta (Blitzcrank)	https://www.reddit.com/r/blitzcrankmains/
BrandMains	Champion Meta (Brand)	https://www.reddit.com/r/BrandMains/
BraumMains	Champion Meta (Braum)	https://www.reddit.com/r/BraumMains/
Caitlynmains	Champion Meta (Caitlyn)	https://www.reddit.com/r/Caitlynmains/
CamilleMains	Champion Meta (Camille)	https://www.reddit.com/r/CamilleMains/
CassiopeiaMains	Champion Meta (Cassiopeia)	https://www.reddit.com/r/CassiopeiaMains/
CorkiMains	Champion Meta (Corki)	https://www.reddit.com/r/CorkiMains/
Dariusmains	Champion Meta (Darius)	https://www.reddit.com/r/Dariusmains/
DianaMains	Champion Meta (Diana)	https://www.reddit.com/r/DianaMains/
DirtySionMains	Champion Meta (Sion)	https://www.reddit.com/r/DirtySionMains/
Draven	Champion Meta (Draven)	https://www.reddit.com/r/Draven/
DrMundoMains	Champion Meta (Dr. Mundo)	https://www.reddit.com/r/DrMundoMains/
ekkomains	Champion Meta (Ekko)	https://www.reddit.com/r/ekkomains/
ezrealmains	Champion Meta (Ezreal)	https://www.reddit.com/r/ezrealmains/
Elisemains	Champion Meta (Elise)	https://www.reddit.com/r/Elisemains/
EvelynnMains	Champion Meta (Evelynn)	https://www.reddit.com/r/EvelynnMains/
FiddlesticksMains	Champion Meta (Fiddlestick)	https://www.reddit.com/r/FiddlesticksMains/
FioraMains	Champion Meta (Fiora)	https://www.reddit.com/r/FioraMains/
fizzmains	Champion Meta (Fizz)	https://www.reddit.com/r/fizzmains/
galiomains	Champion Meta (Galio)	https://www.reddit.com/r/galiomains/
gangplankmains	Champion Meta (Gangplank)	https://www.reddit.com/r/gangplankmains/
GarenMains	Champion Meta (Garen)	https://www.reddit.com/r/GarenMains/
GnarMains	Champion Meta (Gnar)	https://www.reddit.com/r/GnarMains/
GragasMains	Champion Meta (Gragas)	https://www.reddit.com/r/GragasMains/
GravesMains	Champion Meta (Graves)	https://www.reddit.com/r/GravesMains/
GwenMains	Champion Meta (Gwen)	https://www.reddit.com/r/GwenMains/
HecarimMains	Champion Meta (Hecarim)	https://www.reddit.com/r/HecarimMains/
HeimerdingerMains	Champion Meta (Heimerdinger)	https://www.reddit.com/r/HeimerdingerMains/

Illaoi	Champion Meta (Illaoi)	https://www.reddit.com/r/Illaoi/
IreliaMains	Champion Meta (Irelia)	https://www.reddit.com/r/IreliaMains/
ivernmains	Champion Meta (Ivern)	https://www.reddit.com/r/ivernmains/
Janna	Champion Meta (Janna)	https://www.reddit.com/r/Janna/
JarvanIVmains	Champion Meta (Jarvan IV)	https://www.reddit.com/r/JarvanIVmains/
Jaxmains	Champion Meta (Jax)	https://www.reddit.com/r/Jaxmains/
jaycemains	Champion Meta (Jayce)	https://www.reddit.com/r/jaycemains/
JhinMains	Champion Meta (Jhin)	https://www.reddit.com/r/JhinMains/
kaisamains	Champion Meta (Kai'Sa)	https://www.reddit.com/r/kaisamains/
KalistaMains	Champion Meta (Kalista)	https://www.reddit.com/r/KalistaMains/
karmamains	Champion Meta (Karma)	https://www.reddit.com/r/karmamains/
karthusmains	Champion Meta (Karthus)	https://www.reddit.com/r/karthusmains/
KassadinMains	Champion Meta (Kassadin)	https://www.reddit.com/r/KassadinMains/
KatarinaMains	Champion Meta (Katarina)	https://www.reddit.com/r/KatarinaMains/
Kaylemains	Champion Meta (Kayle)	https://www.reddit.com/r/Kaylemains/
KaynMains	Champion Meta (Kayn)	https://www.reddit.com/r/KaynMains/
Kennenmains	Champion Meta (Kennen)	https://www.reddit.com/r/Kennenmains/
KhaZixMains	Champion Meta (Kha'Zix)	https://www.reddit.com/r/KhaZixMains/
Kindred	Champion Meta (Kindred)	https://www.reddit.com/r/Kindred/
KledMains	Champion Meta (Kled)	https://www.reddit.com/r/KledMains/
KogMawMains	Champion Meta (Kog'Maw)	https://www.reddit.com/r/KogMawMains/
KSanteMains	Champion Meta (K'Sante)	https://www.reddit.com/r/KSanteMains/
leagueofjinx	Champion Meta (Jinx)	https://www.reddit.com/r/leagueofjinx/
LeBlancMains	Champion Meta (LeBlanc)	https://www.reddit.com/r/LeBlancMains/
LeeSinMains	Champion Meta (Lee Sin)	https://www.reddit.com/r/LeeSinMains/
LilliaMains	Champion Meta (Lillia)	https://www.reddit.com/r/LilliaMains/
LissandraMains	Champion Meta (Lissandra)	https://www.reddit.com/r/LissandraMains/
LucianMains	Champion Meta (Lucian)	https://www.reddit.com/r/LucianMains/
lulumains	Champion Meta (Lulu)	https://www.reddit.com/r/lulumains/
lux	Champion Meta (Lux)	https://www.reddit.com/r/lux/
malphitemains	Champion Meta (Malphite)	https://www.reddit.com/r/malphitemains/
MalzaharMains	Champion Meta (Malzahar)	https://www.reddit.com/r/MalzaharMains/
MaokaiMains	Champion Meta (Maokai)	https://www.reddit.com/r/MaokaiMains/
MilioMains	Champion Meta (Milio)	https://www.reddit.com/r/MilioMains/
MissFortuneMains	Champion Meta (Miss Fortune)	https://www.reddit.com/r/MissFortuneMains/
MordekaiserMains	Champion Meta (Mordekaiser)	https://www.reddit.com/r/MordekaiserMains/
MorganaMains	Champion Meta (Morgana)	https://www.reddit.com/r/MorganaMains/
NamiMains	Champion Meta (Nami)	https://www.reddit.com/r/NamiMains/
nasusmains	Champion Meta (Nasus)	https://www.reddit.com/r/nasusmains/
NautilusMains	Champion Meta (Nautilus)	https://www.reddit.com/r/NautilusMains/
neekomains	Champion Meta (Neeko)	https://www.reddit.com/r/neekomains/
NidaleeMains	Champion Meta (Nidalee)	https://www.reddit.com/r/NidaleeMains/
NilahMains	Champion Meta (Nilah)	https://www.reddit.com/r/NilahMains/
nocturnemains	Champion Meta (Nocturne)	https://www.reddit.com/r/nocturnemains/
nunumains	Champion Meta (Nunu)	https://www.reddit.com/r/nunumains/
Olafmains	Champion Meta (Olaf)	https://www.reddit.com/r/Olafmains/
OriannaMains	Champion Meta (Orianna)	https://www.reddit.com/r/OriannaMains/
ornnmains	Champion Meta (Orn)	https://www.reddit.com/r/ornnmains/
PantheonMains	Champion Meta (Pantheon)	https://www.reddit.com/r/PantheonMains/
PoppyMains	Champion Meta (Poppy)	https://www.reddit.com/r/PoppyMains/

pykemains	Champion Meta (Pyke)	https://www.reddit.com/r/pykemains/
QiyanaMains	Champion Meta (Qiyana)	https://www.reddit.com/r/QiyanaMains/
QuinnMains	Champion Meta (Quinn)	https://reddit.com/r/QuinnMains/
ChoGathMains	Champion Meta (Cho'Gath)	https://www.reddit.com/r/ChoGathMains/
RakanMains	Champion Meta (Rakan)	https://www.reddit.com/r/RakanMains/
RammusMains	Champion Meta (Rammus)	https://www.reddit.com/r/RammusMains/
reksaimains	Champion Meta (Rek'Sai)	https://www.reddit.com/r/reksaimains/
RellMains	Champion Meta (Rell)	https://www.reddit.com/r/RellMains/
RenataMains	Champion Meta (Renata)	https://www.reddit.com/r/RenataMains/
RenektonMains	Champion Meta (Renekton)	https://www.reddit.com/r/RenektonMains/
Rengarmains	Champion Meta (Rengar)	https://www.reddit.com/r/Rengarmains/
Rivenmains	Champion Meta (Riven)	https://www.reddit.com/r/Rivenmains/
Rumblemains	Champion Meta (Rumble)	https://www.reddit.com/r/Rumblemains/
RyzeMains	Champion Meta (Ryze)	https://www.reddit.com/r/RyzeMains/
SamiraMains	Champion Meta (Samira)	https://www.reddit.com/r/SamiraMains/
sejuanimains	Champion Meta (Sejuani)	https://www.reddit.com/r/sejuanimains/
sennamains	Champion Meta (Senna)	https://www.reddit.com/r/sennamains/
SeraphineMains	Champion Meta (Seraphine)	https://www.reddit.com/r/SeraphineMains/
settmains	Champion Meta (Sett)	https://www.reddit.com/r/settmains/
shacomains	Champion Meta (Shaco)	https://www.reddit.com/r/shacomains/
Shen	Champion Meta (Shen)	https://www.reddit.com/r/Shen/
shyvanamains	Champion Meta (Shyvana)	https://www.reddit.com/r/shyvanamains/
singedmains	Champion Meta (Singed)	https://www.reddit.com/r/singedmains/
Sivir	Champion Meta (Sivir)	https://www.reddit.com/r/Sivir/
SkarnerMains	Champion Meta (Skarner)	https://www.reddit.com/r/SkarnerMains/
sonamains	Champion Meta (Sona)	https://www.reddit.com/r/sonamains/
SorakaMains	Champion Meta (Soraka)	https://www.reddit.com/r/SorakaMains/
SwainMains	Champion Meta (Swain)	https://www.reddit.com/r/SwainMains/
sylasmains	Champion Meta (Sylas)	https://www.reddit.com/r/sylasmains/
syndramains	Champion Meta (Syndra)	https://www.reddit.com/r/syndramains/
Talonmains	Champion Meta (Talon)	https://www.reddit.com/r/Talonmains/
Tahmkenchmains	Champion Meta (Tahm Kench)	https://www.reddit.com/r/Tahmkenchmains/
taricmains	Champion Meta (Taric)	https://www.reddit.com/r/taricmains/
TeemoTalk	Champion Meta (Teemo)	https://www.reddit.com/r/TeemoTalk/
thesecretweapon	Champion Meta (Zac)	https://www.reddit.com/r/thesecretweapon/
ThreshMains	Champion Meta (Thresh)	https://www.reddit.com/r/ThreshMains/
TristanaMains	Champion Meta (Tristana)	https://www.reddit.com/r/TristanaMains/
Trundlemains	Champion Meta (Trundle)	https://www.reddit.com/r/Trundlemains/
TryndamereMains	Champion Meta (Tryndamere)	https://www.reddit.com/r/TryndamereMains/
TwistedFateMains	Champion Meta (Twisted Fate)	https://www.reddit.com/r/TwistedFateMains/
TwitchMains	Champion Meta (Twitch)	https://www.reddit.com/r/TwitchMains/
Udyrmains	Champion Meta (Udyr)	https://www.reddit.com/r/Udyrmains/
UrgotMains	Champion Meta (Urgot)	https://www.reddit.com/r/UrgotMains/
Velkoz	Champion Meta (Vel'Koz)	https://www.reddit.com/r/Velkoz/
vexmains	Champion Meta (Vex)	https://www.reddit.com/r/vexmains/
ViMains	Champion Meta (Vi)	https://www.reddit.com/r/ViMains/
viktormains	Champion Meta (Viktor)	https://www.reddit.com/r/viktormains/
VladimirMains	Champion Meta (Vladimir)	https://www.reddit.com/r/VladimirMains/
VolibearMains	Champion Meta (Volibear)	https://www.reddit.com/r/VolibearMains/
warwickmains	Champion Meta (Warwick)	https://www.reddit.com/r/warwickmains/

Wukongmains	Champion Meta (Wukong)	https://www.reddit.com/r/Wukongmains/
xayahmains	Champion Meta (Xayah)	https://www.reddit.com/r/xayahmains/
XerathMains	Champion Meta (Xerath)	https://www.reddit.com/r/XerathMains/
XinZhaoMains	Champion Meta (Xin Zhao)	https://www.reddit.com/r/XinZhaoMains/
YasuoMains	Champion Meta (Yasuo)	https://www.reddit.com/r/YasuoMains/
YIMO	Champion Meta (Master Yi)	https://www.reddit.com/r/YIMO/
YoneMains	Champion Meta (Yone)	https://www.reddit.com/r/YoneMains/
yorickmains	Champion Meta (Yorick)	https://www.reddit.com/r/yorickmains/
yuumimains	Champion Meta (Yuumi)	https://www.reddit.com/r/yuumimains/
vaynemains	Champion Meta (Vayne)	https://www.reddit.com/r/vaynemains/
VarusMains	Champion Meta (Varus)	https://www.reddit.com/r/VarusMains/
VeigarMains	Champion Meta (Veigar)	https://www.reddit.com/r/VeigarMains/
ViegoMains	Champion Meta (Viego)	https://www.reddit.com/r/ViegoMains/
zedmains	Champion Meta (Zed)	https://www.reddit.com/r/zedmains/
ZeriMains	Champion Meta (Zeri)	https://www.reddit.com/r/ZeriMains/
ZiggsMains	Champion Meta (Ziggs)	https://www.reddit.com/r/ZiggsMains/
ZileanMains	Champion Meta (Zilean)	https://www.reddit.com/r/ZileanMains/
zoemains	Champion Meta (Zoe)	https://www.reddit.com/r/zoemains/
zyramains	Champion Meta (Zyra)	https://www.reddit.com/r/zyramains/

C Total Number of collected Posts per Subreddit



D League of Legends Patch Version Release Dates

Patch Version	Release Date
14.9	2024-04-30
14.8	2024-04-17
14.7	2024-04-03
14.6	2024-03-20
14.5	2024-03-06
14.4	2024-02-22
14.3	2024-02-07
14.2	2024-01-24
14.1	2024-01-10

E Duplicate Terms and Quantities for TF-IDF Term Clusters of Reddit Post Contents

index	k	duplicates	duplicates_per	duplicates_top
0	20	174	0.870000	like: 18, know: 13, game: 10, think: 9, really: 9
1	19	164	0.863158	like: 17, know: 11, game: 10, think: 9, build: 8
2	18	153	0.850000	like: 16, play: 11, know: 10, think: 10, game: 8
3	17	140	0.823529	like: 15, play: 11, think: 10, know: 9, game: 7
4	16	131	0.818750	like: 14, play: 10, know: 9, think: 9, really: 7
5	15	121	0.806667	like: 13, play: 11, think: 8, know: 8, build: 7
6	14	113	0.807143	like: 12, play: 9, know: 8, good: 8, think: 7
7	13	107	0.823077	like: 11, play: 9, good: 8, know: 7, really: 7
8	12	98	0.816667	like: 10, play: 8, good: 8, really: 7, think: 6
9	11	88	0.800000	like: 9, play: 7, good: 7, really: 7, know: 6
10	10	78	0.780000	like: 8, know: 6, game: 5, damage: 5, playing: 5
11	9	66	0.733333	like: 7, know: 7, build: 5, think: 5, game: 4
12	8	63	0.787500	like: 6, play: 6, build: 5, know: 5, game: 4
13	7	52	0.742857	like: 6, think: 5, damage: 4, new: 4, good: 4
14	6	42	0.700000	like: 5, know: 5, damage: 4, think: 4, good: 4
15	5	34	0.680000	like: 4, know: 4, game: 3, think: 3, damage: 3
16	4	26	0.650000	know: 4, like: 3, think: 3, build: 2, game: 2
17	3	18	0.600000	like: 3, game: 3, build: 2, damage: 2, know: 2

F League of Legends Item Dictionary


```
{"Cull": ["Cull", "Cul", "Culll"], "Dark Seal": ["Dark
   Seal", "Dark-Seal", "Dark-Seel", "Dark Seel"], "Dorans
    Blade": ["Doran's Blade", "Dorans Blade", "Dorans
   Blad", "Doran's Blad", "D Blade", "D-Blade", "DBlade"
   ], "Dorans Ring": ["Doran's Ring", "Dorans Ring", "D
   Ring", "D-Ring", "DRing"], "Dorans Shield": ["Doran's
   Shield", "Dorans Shield", "D Shield", "D-Shield", "
   DShield"], "Gustwalker Hatchling": ["Gustwalker-
   Hatchling", "Gustwalker", "Hatchling", "Gastwalker", "
   Blue-Smite", "Blue Smite", "BlueSmite"], "Mosstomper
   Seedling": ["Mosstomper-Seedling", "Mosstomper", "
   Seedling", "Mostomper", "Green-Smite", "Green Smite",
   "GreenSmite"], "Scorchclaw Pup": ["Scorchclaw-Pup", "
   Scorchclaw", "Pup", "Skorchclaw", "Red-Smite", "Red
   Smite", "RedSmite"], "Tear of the Goddess": ["Tear of
   the Goddess", "Tear of Goddess", "Tear", "Mana-Tear",
   "Mana Tear", "Goddess Tear", "Tear of the Godess", "
   Tear of the Goddes"], "World Atlas": ["World Atlas", "
   World-Atlas", "World Attlas", "World-Attlas"], "
   Control Ward": ["Control Ward", "Control-Ward"], "
   Elixir of Iron": ["Elixir of Iron", "Elexir of Iron",
   "Iron Elixir", "Iron-Elixir", "Yellow Elixir"], "
   Elixir of Sorcery": ["Elixir of Sorcery", "Elexir of
   Sorcery", "Sorcery Elixir", "Sorcery-Elixir", "Blue
   Elixir", "Purple Elixir"], "Elixir of Wrath": ["Elixir
    of Wrath", "Elexir of Wrath", "Wrath Elixir", "Wrath-
   Elixir", "Elixir of Warth", "Warth Elixir", "Warth-
   Elixir", "Red Elixir"], "Health Potion": ["Health
   Potion", "Health-Potion", "Heeling Potion", "Red
   Potion", "Red-Potion", "Health Potions", "Health-
   Potions", "Heeling Potions", "Red Potions"], "
   Refillable Potion": ["Refillable Potion", "Refillable-
   Potion", "Refillable", "Refillables", "Refillable
   Potions", "Refillable-Potions"], "Farsight Alteration"
   : ["Farsight", "Alteration", "Farsight-Alteration", "
   Blue Trinket", "Blue-Trinket", "BlueTrinket", "Blue
   Ward", "Blue-Ward", "BlueWard", "Blue Wards"], "Oracle
    Lens": ["Oracle Lens", "Oracle Lense", "Oracle-Lense"
   , "Oracle-Lens", "Oracle", "Sweeper", "Sweeper-Drone",
    "SweeperDrone", "Sweaper", "Sweeper Drone"], "Stealth
    Ward": ["Stealth Ward", "Stealth-Ward", "Totem Ward",
```

```
 "Yellow Ward", "Yellow Wards", "Yellow-Ward", "Yellow
-Wards", "Stealth Wards", "Stealth-Wards"], "Slightly
Magical Boots": ["Slightly Magical Boots", "Slightly-
Magical Boots", "Slightly-Magical-Boots", "Magical
Boots", "Magical-Boots", "Magic Boots"], "Biscuit": ["
Biscuit", "Biscuits", "Total Biscuit", "Total Biscuit
of Everlasting Will", "Biscuitts", "Total-Biscuit", "
Total-Biscuits"], "Elixir of Avarice": ["Avarice", "
Avarice-Elixir"], "Elixir of Force": ["Elixir of Force
", "Force-Elixir", "Elixir-of-Force"], "Elixir of
Skill": ["Elixir of Skill", "Skill-Elixir", "Elixir-of
-Skill"], "Berserkers Greaves": ["Berserker's", "
Greaves", "Berserkers", "Berserkers-Greaves", "
Berserks", "Berserk's", "Attack Speed Boots", "Beserks
", "Beserk's"], "Boots": ["Boots", "Standard-Boots", "
Basic-Boots", "Starter-Boots", "Tier-One-Boots", "Tier
-1-Boots"], "Boots of Swiftness": ["Boots of Swiftness
", "Swiftness-Boots", "Boots of Swiftnes", "Swiftnes-
Boots", "Swifties", "Swiftys", "Swiftness Boots"], "
Boots of Lucidity": ["Ionian Boots", "Boots of
Lucidity", "Lucidity-Boots", "Lucids", "Lucid's", "
Ionians", "Ionian's"], "Mercurys Treads": ["Mercury's"
, "Mercury", "Mercs", "Merks", "Merc", "Treads", "MR
Boots", "MR-Boots", "Merc-Treads", "Tenacity Boots", "
Tenacity-Boots"], "Plated Steelcaps": ["Plated", "
Steelcaps", "Plated-Steelcaps", "Tabbies", "Tabbys", "
Ninja-Tabbies", "Ninja-Tabi", "Tabi", "Tabies"], "
Sorcerers Shoes": ["Sorcerer's Shoes", "Sorcerers
Shoes", "Sorc Shoes", "Sorc-Shoes", "Sorcs", "MP Boots
", "Magicpen Boots", "MPen Boots", "M-Pen Boots"], "
Symbiotic Soles": ["Symbiotic Soles", "Symbiotic-Soles
", "Symbiotic Boots", "Symbiotic-Boots"], "Zephyr": ["
Zephyr", "Zephyr-Boots"], "Amplifying Tome": ["
Amplifying Tome", "Amplifying-Tome", "Amp Tome", "Amp-
Tome", "Tome", "Tomes", "Amp-Tomes", "Amp Tomes"], "BF
 Sword": ["B. F. Sword", "BF Sword", "BF-Sword", "Big
Fucking Sword", "BFS"], "Blasting Wand": ["Blasting
Wand", "Blasting-Wand", "BWand", "Blast-Wand", "Blast
Wand", "B-Wand"], "Cloak of Agility": ["Cloak of
Agility", "Agility Cloak", "Agility-Cloak", "Cloak", "
Crit-Cloak", "Cloaks"], "Cloth Armor": ["Cloth Armor",
 "Cloth-Armor", "Amor Cloth", "Amor-Cloth", "Amored
Cloth", "Cloth"], "Dagger": ["Dagger", "Dager", "
```

```
Attack-Speed-Dagger", "AS-Dagger"], "Faerie Charm": ["
Faerie Charm", "Faerie-Charm", "Faery Charm", "Faeries
 Charm", "Ferry Charm", "Fearicharm"], "Glowing Mote":
 ["Glowing Mote", "Glowing-Mote", "Mote", "Motte", "
Haste Mote", "Haste-Mote", "Motes"], "Long Sword": ["
Long Sword", "Long-Sword", "Longsword"], "Needlessly
Large Rod": ["Needlessly-Large Rod", "Large Rod", "
Needlessly-Large-Rod", "NLR", "Large Rods"], "Null
Magic Mantle": ["Null-Magic Mantle", "Null Magic
Mantle", "Nullmagic Mantle", "Nullmagic-Mantle", "
Mantle", "MR-Mantle", "Mantles"], "Pickaxe": ["Pickaxe
", "Pickaxes", "Pick-Axe", "Pick-Axes"], "Rejuvenation
 Bead": ["Rejuvenation Bead", "Rejuvenation-Bead", "
RBead", "R-Bead", "Bead of Rejuvenation", "
Rejuvenation Beads", "R-Beads"], "Ruby Crystal": ["
Ruby Crystal", "Ruby-Crystal", "Ruby", "Rubies", "HP-
Crystal"], "Sapphire Crystal": ["Sapphire Crystal", "
Sapphire-Crystal", "Sapphire", "Sapphires", "Saphire
Crystal", "Sapphire-Crystals", "Mana Crystal", "Mana
Crystals"], "Aether Wisp": ["Aether", "Aether-Wisp", "
Wisp", "Wisps"], "Bamis Cinder": ["Bamis-Cinder", "
Bamis", "Bami's", "Bami's-Cinder", "Cinder"], "
Bandleglass Mirror": ["Bandleglass", "Bandleglass-
Mirror", "Bandle Mirror", "Bandle-Mirror"], "Blighting
 Jewel": ["Blighting Jewel", "Blighting-Jewel", "
Blight Jewel", "Blight-Jewel"], "Bramble Vest": ["
Bramble-Vest", "Bramble"], "Catalyst of Aeons": ["
Catalyst", "Aeons-Catalyst", "Aeon-Catalyst", "
Catalysts"], "Caulfields Warhammer": ["Caulfield's", "
Warhammer", "Caulfields", "Caulefields", "Caulefield's
", "Warhammers"], "Chain Vest": ["Chain Vest", "Chain-
Vest", "Armor Vest", "Armor-Vest"], "Crystalline
Bracer": ["Crystalline", "Crystalline-Bracer", "
Crystalline-Bracers", "Bracer", "Bracers", "Crystal-
Bracers"], "Executioners Calling": ["Executioner's", "
Executioners", "Calling", "Executioners-Calling", "
Executioner's-Calling"], "Fated Ashes": ["Fated-Ashes"
, "Fated Ashes", "Fated Ashe"], "Fiendish Codex": ["
Fiendish", "Fiendish-Codex", "Codex", "Codexes", "
Codices"], "Forbidden Idol": ["Forbidden Idol", "
Forbidden-Idol", "Forbiden Idol", "Forbiden-Idol"], "
Giants Belt": ["Giant's Belt", "Giant's-Belt", "Giants
-Belt", "Giants Belt", "Giant Belt", "Giant-Belt"], "
```

```
Glacial Buckler": ["Glacial Buckler", "Glacial-Buckler
", "Glacier Buckler", "Glacier-Buckler"], "Haunting
Guise": ["Haunting Guise", "Haunting-Guise"], "
Hearthbound Axe": ["Hearthbound Axe", "Hearthbound-Axe
", "Heartbound Axe", "Heartbound-Axe"], "Hexdrinker":
["Hexdrinker", "Hex Drinker", "Hex-Drinker"], "Hextech
 Alternator": ["Hextech-Alternator", "Alternator", "
Alternators", "Hex-Tech-Alternator"], "Kindlegem": ["
Kindlegem", "Kindle Gem", "Kindle-Gem", "Kindelinggem"
, "Kindeling-Gem", "Kindling Gem"], "Last Whisper": ["
Last Whisper", "Last-Whisper"], "Lost Chapter": ["Lost
 Chapter", "Lost-Chapter", "Lostchapter"], "Negatron
Cloak": ["Negatron", "Negatron-Cloak", "MR-Cloal", "MR
 Cloak"], "Noonquiver": ["Noonquiver", "Noon Quiver",
"Noon-Quiver"], "Oblivion Orb": ["Oblivion Orb", "
Oblivion-Orb", "Obliv-Orb", "Obliv Orb", "Ob-Orb", "
Oblivionorb"], "Phage": ["Phage", "Phag", "Phague"], "
Quicksilver Sash": ["Quicksilver", "Sash", "
Quicksilver-Sash", "QSS", "Q.S.S", "QSS"], "Rectrix":
["Rectrix", "Recktrix", "Rectricks"], "Recurve Bow": [
"Recurve Bow", "Recurve-Bow", "Recurvebow"], "Runic
Compass": ["Runic Compass", "Runic-Compass", "
Runiccompass", "Runicompass", "Runic Compas"], "Scouts
 Slingshot": ["Scout's", "Scouts", "Scout's-Slingshot"
, "Scouts-Slingshot", "Slingshot", "Slingshots"], "
Seekers Armguard": ["Seeker's", "Armguard", "Seekers",
 "Seekers-Armguard", "Seeker's-Armguard", "Shattered-
Armguard"], "Serrated Dirk": ["Dirks", "Serrated-Dirk"
, "Dirk"], "Sheen": ["Sheen", "Sheens"], "Spectress
Cowl": ["Spectre's Cowl", "Spectres Cowl", "Spectre's-
Cowl", "SpectresCowl", "Spectrescowl"], "Steel Sigil":
 ["Steel-Sigil", "Sigil", "Sigils"], "Brutalizer": ["
Brutalizer", "Brutealizer"], "Tiamat": ["Tiamat", "
Tiammat", "Tiamate"], "Tunneler": ["Tunneler", "
Tunnelers", "Tuneler", "Tunneller"], "Vampiric Scepter
": ["Vampiric", "Vampiric-Scepter", "Scepter", "
Scepters"], "Verdant Barrier": ["Verdant Barrier", "
Verdant-Barrier"], "Wardens Mail": ["Warden's Mail", "
Wardens Mail", "Warden's-Mail", "Wardens-Mail", "
Warden Mail", "Warden-Mail"], "Watchful Wardstone": ["
Watchful Wardstone", "Watchful-Wardstone"], "Winged
Moonplate": ["Moonplate", "Winged-Moonplate", "
Moonplates"], "Zeal": ["Zeal", "Zeals"], "Abyssal Mask
```

```
": ["Abyssal", "Abyssal-Mask"], "Archangels Staff": ["
Archangel's", "Archangels", "Archangels-Staff", "
Archangel-Staff", "Arch-Staff", "Arch Staff", "Arc-
Staff", "Arc Staff"], "Ardent Censer": ["Ardent", "
Ardent-Censer", "Ardend"], "Axiom Arc": ["Axiom", "
Axiom-Arc", "Axxiom", "Axiome", "Axxiome"], "Banshees
Veil": ["Banshees", "Banshees-Veil", "Banshee's", "
Banshes", "Banshe's", "Bannshees"], "Black Cleaver": [
"Cleaver", "Black-Cleaver"], "Blackfire Torch": ["
Blackfire", "Blackfire-Torch", "Black-Fire-Torch"], "
BORK": ["Blade of the Ruined King", "Blade of Ruined
King", "Ruined King Blade", "BORK", "B.O.R.K", "BORG",
 "Blade of the Ruined"], "Bloodsong": ["Bloodsong", "
Blood-Song", "Blodsong"], "Bloodthirster": ["
Bloodthirster", "Blood-Thirster-Song"], "Bounty of
Worlds": ["Bounty of Worlds", "Bounty of World", "
Bounty Worlds", "Bounty-of-Worlds"], "Celestial
Opposition": ["Celestial Opposition", "Celestial-
Opposition", "Celestial Oposition", "Celestial-
Oposition"], "Chempunk Chainsword": ["Chainsword", "
Chempunk-Chainsword", "Chain-Sword"], "Cosmic Drive":
["Cosmic Drive", "Cosmic-Drive"], "Cryptbloom": ["
Cryptbloom", "Crypt-Bloom", "Cryptboom", "Cryptobloom"
, "Crypto-Bloom"], "Dawncore": ["Dawncore", "Dawn-Core
"], "Dead Mans Plate": ["Dead Man's", "Dead Mans", "
Dead-Man's-Plate", "Dead-Mans-Plate"], "Deaths Dance":
 ["Deaths Dance", "Death's Dance", "Death's-Dance", "
Deaths-Dance"], "Dream Maker": ["Dream Maker", "Dream-
Maker"], "Echoes of Helia": ["Echoes of Helia", "
Echoes-of-Helia", "Echoes-Helia", "Helia Echoes", "
Helia-Echoes", "Echos of Helia", "Echos-of-Helia"], "
Eclipse": ["Eclipse", "Eclips"], "Edge of Night": ["
Edge of Night", "Edge-of-Night", "Edge Night"], "
Essence Reaver": ["Essence Reaver", "Essence-Reaver",
"Esence Reaver", "Essence Reever"], "Experimental
Hexplate": ["Hexplate", "Experimental-Hexplate", "
Hexxplate", "Hexplatte"], "Fimbulwinter": ["
Fimbulwinter", "Fimbelwinter", "Fimbul-Winter", "
Fimblewinter", "Fimble-Winter", "Fimbul"], "Force of
Nature": ["Force of Nature", "Force-of-Nature", "Force
 Nature", "Force-Nature"], "Frozen Heart": ["Frozen
Heart", "Frozen-Heart"], "Guardian Angel": ["Guardian"
, "Guardian-Angel", "Guardians", "Guardians-Angel", "
```

```
GA", "G.A."], "Guinsoos Rageblade": ["Guinsoo's", "
Guinsoos", "Rageblade", "Guinsoo's-Rageblade", "
Guinsoos-Rageblade", "Guinsos", "Guinso's", "Rage-
Blade"], "Heartsteel": ["Heartsteel", "Heart-Steel", "
Heartsteal", "Heart-Steal"], "Hextech Rocketbelt": ["
Hextech-Rocketbelt", "Rocketbelt", "Hex-Belt", "
Hexbelt", "Hextech Belt", "Hextech-Belt", "Rocket Belt
", "Rocket-Belt"], "Hollow Radiance": ["Hollow
Radiance", "Hollow-Radiance", "Hallow Radiance"], "
Horizon Focus": ["Horizon Focus", "Horizon-Focus"], "
Hubris": ["Hubris", "Hubriss", "Houbris"], "
Hullbreaker": ["Hullbreaker", "Hull-Breaker", "
Hullbraker", "Hull-Braker"], "Iceborn Gauntlet": ["
Iceborn", "Iceborn-Gauntlet", "Ice-Born"], "Immortal
Shieldbow": ["Shieldbow", "Immortal-Shieldbow", "
Shield-Bow"], "Imperial Mandate": ["Imperial", "
Mandate", "Imperial-Mandate"], "Infinity Edge": ["
Infinity Edge", "Infinity-Edge", "IE", "I.E.", "
Infinitys Edge", "Infinities Edge", "Infinity's Edge"
], "Jaksho": ["Jaksho", "Jak'Sho", "Jak-Sho", "Protean
", "Jacksho", "Jack-Sho", "Jack'Sho"], "Kaenic Rookern
": ["Kaenic", "Rookern", "Kaenic-Rookern"], "Knights
Vow": ["Knight's Vow", "Knight's-Vow", "Knights Vow",
"Knights-Vow"], "Kraken Slayer": ["Kraken", "Krakens",
 "Kraken's", "Kraken-Slayer"], "Liandrys Torment": ["
Liandry's", "Liandrys", "Liandries", "Liandry's-
Torment", "Liandrys-Torment"], "Lich Bane": ["Lich-
Bane", "Lichbane", "Lichban", "Lich Bane"], "Locket":
["Locket", "Iron Solari", "Locket-of-the-Iron-Solari",
 "Lockett"], "Lord Dominiks Regards": ["Lord Dominik's
", "Lord Dominiks", "Lord-Dominiks", "Lord-Dominik's",
 "LDR", "L.D.R.", "L.D.R", "Lord-Dominiks-Regards"], "
Ludens Companion": ["Luden's", "Ludens", "Ludens-
Companion", "Luden's-Companion"], "Malignance": ["
Malignance", "Malignace", "Mallignance", "Mallignace"
], "Manamune": ["Manamune", "Mana-Mune", "Mannamune",
"Manamun"], "Maw": ["Maw", "Maw's", "Maws", "
Malmortius"], "Mejais Soulstealer": ["Mejai's", "
Mejais", "Mejas", "Meja", "Soulstealer", "Mejais-
Soulstealer", "Mejai's-Soulstealer"], "Mercurial
Scimitar": ["Scimitar", "Mercurial-Scimitar", "
Mercurial"], "Mikaels Blessing": ["Mikaels", "Mikael's
", "Mikaels-Blessing", "Mikael's-Blessing"], "
```

```
Moonstone Renewer": ["Moonstone", "Moonstone-Renewer",
 "Moon-Stone"], "Morellonomicon": ["Morellonomicon", "
Morello", "Morellos", "Morello's", "Morelonomicon", "
Morellonomikon", "Morelonomikon"], "Mortal Reminder":
["Mortal Reminder", "Mortal-Reminder"], "Muramana": ["
Muramana", "Muramanma", "Muramanma", "Mura-Mana"], "
Nashors Tooth": ["Nashor's-Tooth", "Nashors", "Nashor'
s", "Nashors-Tooth"], "Navori Flickerblade": ["
Flickerblade", "Flicker-Blade", "Navori-Flickerblade"
], "Opportunity": ["Opportunity", "Oportunity"], "
Overlords Bloodmail": ["Bloodmail", "Blood-Mail", "
Overlord's", "Overlords"], "Phantom Dancer": ["Phantom
 Dancer", "Phantom-Dancer"], "Profane Hydra": ["
Profane", "Profane-Hydra", "Proffane"], "Rabadons
Deathcap": ["Rabadon's", "Rabadons", "Rabbadons", "
Rabaddons", "Deathcap", "Death-Cap", "Rabadons-
Deathcap"], "Randuin's Omen": ["Randuin's", "Randuins"
, "Randuin", "Randuin's-Omen", "Randuins-Omen"], "
Rapid Firecannon": ["Firecannon", "Rapid-Firecannon",
"Fire-Cannon"], "Ravenous Hydra": ["Ravenous", "
Ravenous-Hydra"], "Redemption": ["Redemption", "
Redemtion"], "Riftmaker": ["Riftmaker", "Rift-Maker",
"Rift Maker"], "Rod of Ages": ["Rod of Ages", "Rod-of-
Ages", "Ages Rod"], "Runaans Hurricane": ["Runaan's",
"Runaans", "Runaans-Hurricane", "Runaan's-Hurricane",
"Runan's", "Runans"], "Rylais Crystal Scepter": ["
Rylai's", "Rylais", "Raylai's", "Raylais", "Crystal
Scepter", "Crystal-Scepter"], "Seraphs Embrace": ["
Seraph's-Embrace", "Seraph's", "Seraphs", "Seraphs-
Embrace"], "Serpents Fang": ["Serpent's", "Serpents",
"Serpents-Fang", "Serpent's-Fang"], "Seryldas Grudge":
 ["Serylda's-Grudge", "Seryldas-Grudge", "Seryldas", "
Serylda's", "Sereldas", "Serelda's"], "Shadowflame": [
"Shadowflame", "Shadow-Flame", "Shaddowflame", "
Shaddow-Flame"], "Shurelyas Battlesong": ["Shurelya's"
, "Shurelyas", "Shurelyas-Battlesong", "Shurelya's-
Battlesong"], "Solstice Sleigh": ["Solstice Sleigh", "
Solstice-Sleigh", "Sol Sleigh", "Sol-Sleigh", "Sol.
Sleigh"], "Spear of Shojin": ["Shojin", "Spear-Shojin"
, "Shojin-Spear", "Spear-of-Shojin"], "Sunfire Aegis":
 ["Sunfire", "Aegis", "Sunfire-Aegis", "Sun-Fire"], "
Spirit Visage": ["Spirit-Visage", "Visage"], "Staff of
 Flowing Water": ["Staff of Flowing Water", "Flowing
```

```
Water Staff", "Flowing-Water-Staff"], "Statikk Shiv":
["Statikk", "Shiv", "Statik Shiv", "Statik-Shiv", "
Statikk-Shiv"], "Steraks Gage": ["Sterak's", "Steraks"
, "Steracks", "Sterak's-Gage", "Steraks-Gage"], "
Stormsurge": ["Stormsurge", "Storm-Surge", "Storm
Surge"], "Stridebreaker": ["Stridebreaker", "Stride-
Breaker", "Stride Breaker", "Stridebraker", "Stride
Braker"], "Sundered Sky": ["Sundered", "Sundered-Sky"
], "Terminus": ["Terminus", "Termimnus", "Termminus",
"Therminus"], "The Collector": ["Collector", "Colector
", "Collecktor"], "Thornmail": ["Thornmail", "Thorn-
Mail", "Thornmaile", "Thorn Mail", "Thornmale"], "
Titanic Hydra": ["Titanic", "Titanic-Hydra", "Titanick
"], "Trailblazer": ["Trailblazer", "Trailblaizer", "
Trail-Blazer", "Trail Blazer"], "Trinity Force": ["
Trinity-Force", "Trinity", "Triforce", "Tri-Force", "
Tri Force"], "Umbral Glaive": ["Umbral-Glaive", "
Umbral"], "Unending Despair": ["Unending Despair", "
Unending-Despair", "Un-ending Despair"], "Vigilant
Wardstone": ["Wardstone", "Ward-Stone", "Vigilant-
Wardstone"], "Void Staff": ["Void Staff", "Void Staf",
 "Void-Staff", "Void-Staf"], "Voltaic Cyclosword": ["
Voltaic", "Cyclosword", "Cyclo-Sword", "Voltaic-
Cyclosword"], "Warmogs Armor": ["Warmog's", "Warmogs",
 "Warmongs", "Warmong's", "Warmogs-Armor"], "Winters
Approach": ["Winter's Approach", "Winters Approach", "
Winters-Approach", "Winter Approach", "Winters Aproach
", "Winter's-Approach", "Winters-Approach"], "Wits End
": ["Wits End", "Wit's End", "Wits-End", "Wit's-End",
"Witsend", "Witts End", "Witt's End"], "Youmuus
Ghostblade": ["Youmuu's", "Youmuus", "Youmus", "Youmu'
s", "Yomus", "Yomu's", "Ghostblade", "Ghost-Blade"], "
Yun Tal Wildarrows": ["Yun Tal", "Yun-Tal", "
Wildarrows", "Wildarrow", "Wild-Arrows", "Wild-Arrow"
], "Zaz'Zak's Realmspike": ["Zaz'Zak's", "Zak's", "
Zaks", "Zack's", "Zacks", "Realmspike", "Realm-Spike"
], "Zhonya's Hourglass": ["Zhonya's", "Zhonyas", "
Zhonya", "Zonya's", "Zonyas", "Zonya", "Hourglass", "
Hour-Glass", "Hourglas"]}
```

G League of Legends Champion Dictionary


```
{"Aatrox": ["World Ender", "Darkin Blade", "Aatrox", "Red
    Sword Guy", "Atrox", "Aatrocks", "Aatroxx", "Atroxes"
   , "Aatrokz", "Aatroks", "Atroxx"], "Ahri": ["Ahri", "
   Nine-Tailed Fox", "Foxy Lady", "Fox Girl", "Fox
   Champion", "Nine Tailed Fox", "Ari", "Ahrri", "Ary", "
   Kitsune", "Fox Spirit"], "Akali": ["Akali", "Akalli",
   "Akkali", "Akaly", "Ackali", "Fist of Shadow"], "
   Akshan": ["Akshan", "Rogue Sentinel", "Ak", "Ashkan",
   "Ackshan", "Aksha", "Ashka", "Akschan", "Ackschan"], "
   Alistar": ["Alistar", "Ali", "Alis", "Minotaur", "
   Allistar", "Alistair", "Allistair", "Alli"], "Amumu":
   ["Amumu", "Mumu", "Amum", "Ammumu", "Amummu", "Ammummu
   ", "Amu", "Mummy", "Sad Mummy"], "Anivia": ["Anivia",
   "Cryophoenix", "Phoenix", "Annivia", "Aniwia", "Aniv",
    "Anniv", "Nivia"], "Annie": ["Dark Child", "Annie", "
   Anni", "Anny", "Anniie", "Anniy", "Anniiee", "Ani", "
   Anne", "Aniie", "Anney", "Annii"], "Aphelios": ["
   Weapon of the Faithful", "Aphelios", "Aphel", "Phelios
   ", "Afelios", "Aphelius", "Apheli", "Phelius"], "Ashe"
   : ["Ashe", "Frost Archer", "Ash", "Asch", "Ashee"], "
   Aurelion": ["Aurelion", "Sol", "Star Forger", "Aurel",
    "Aurellion", "Aurrelion"], "Azir": ["Azir", "Emperor
   of Sands", "Emperor of the Sands", "Asir", "Azzir", "
   Azire"], "Bard": ["Bard", "Wandering Caretaker", "Bart
   ", "Barde"], "Belveth": ["Bel'Veth", "Belveth", "Void
   Empress", "Empress of the Void", "Belvef", "Bellveth",
    "Bell'Veth", "Belleveth", "Bel Veth", "Belle Veth", "
   Bel Veht"], "Blitzcrank": ["Blitzcrank", "Steam Golem"
   , "Great Steam Golem", "Blitz", "Crank", "Blizcrank",
   "Blitzkrank"], "Brand": ["Brand", "Burning Vengeance",
    "Barnd", "Brande", "Brend", "Brandt"], "Braum": ["
   Braum", "Heart of the Freljord", "Heart of Freljord",
   "Braun", "Browm", "Braumm"], "Briar": ["Briar", "
   Restrained Hunger", "Briair", "Breiar", "Breear", "
   Bria"], "Caitlyn": ["Caitlyn", "Sheriff of Piltover",
   "Kaitlyn", "Cait", "Kait", "Catelyn", "Katelym", "
   Caitlin", "Kaitlin", "Caitlynn", "Catelynn"], "Camille
   ": ["Camille", "Steel Shadow", "Steel Shaddow", "
   Camile", "Camil", "Camill", "Kamil", "Kamille", "
   Camilie", "Camill"], "Cassiopeia": ["Cassiopeia", "
   Serpents Embrace", "Serpent's Embrace", "Serpent
```

```
Embrace", "Cass", "Cassio", "Casio", "Casiopeia"], "
Chogath": ["Chogath", "Cho'Gath", "Terror of the Void"
, "Terror of Void", "Cho", "Chogat"], "Corki": ["Corki
", "Daring Bombardier", "Korki", "Corky", "Korcky", "
Corcki"], "Darius": ["Darius", "Hand of Noxus", "
Darios", "Darrius"], "Diana": ["Diana", "Scorn of the
Moon", "Scorn of Moon", "Dianna"], "Mundo": ["Mundo",
"Dr. Mundo", "Madman of Zaun", "Munndo", "Doktor Mundo
", "Doctor Mundo"], "Draven": ["Draven", "Glorious
Executioner", "Dravenn", "Dreven", "Drawen"], "Ekko":
["Ekko", "Boy who shattered Time", "Echo", "Ecko", "
Ecco"], "Elise": ["Elise", "Spider Queen", "Elize", "
Elis", "Eliz"], "Evelynn": ["Evelynn", "Agony's
Embrace", "Agonys Embrace", "Agony Embrace", "Evelin",
 "Eve", "Evelinn", "Evelyn", "Evlyn", "Evlin"], "
Ezreal": ["Ezreal", "Prodigal Explorer", "Ezrael", "
Esreal", "Esrael"], "Fiddlesticks": ["Fiddlesticks", "
Ancient Fear", "Fiddle", "Fiddel", "Fiddlestix", "
Fiddelsticks", "Fiddelstix", "Fiddle Stix", "Fiddle
Sticks"], "Fiora": ["Fiora", "Grand Duelist", "Fiona",
 "Fjora"], "Fizz": ["Fizz", "Tidal Trickster", "Fiz",
"Fitz"], "Galio": ["Galio", "ColossusGallio", "Galios"
, "Gallios"], "Gangplank": ["Gangplank", "Saltwater
ScourgeGankplank", "Gangplanck", "Plank", "Gankplanck"
], "Garen": ["Garen", "Might of DemaciaGarren", "Geren
", "Gerren"], "Gnar": ["Gnar", "Missing LinkGnarr"], "
Gragas": ["Gragas", "Rabble RouserGraggas", "Grags"],
"Graves": ["Graves", "The OutlawGravse"], "Gwen": ["
Gwen", "Hallowed SeamstressGven", "Gewn", "Gvenn", "
Gwenn"], "Hecarim": ["Hecarim", "Shadow of War", "
Shaddow of War", "Hekarim", "Heckarim", "Heccarim", "
Hecarimm", "Heckarimm", "Heccarimm", "Heca", "Hecca",
"Hecka"], "Heimerdinger": ["Heimerdinger", "Revered
Inventor", "Heimer", "Dinger", "Heymerdinger", "Heymer
", "Heimeding", "Heimeding", "Heimdinge", "Heimdinger"
], "Hwei": ["Hwei", "The Visionary", "Huwei", "Huwai",
 "Hwai", "Hway", "Hwey", "Huway", "Huwey"], "Illaoi":
["Illaoi", "Kraken Priestess", "Ilaoi", "Lllaoi", "
Iloi", "Illoi", "Ilao", "Illao", "Illoy", "Illaoy"], "
Irelia": ["Irelia", "Blade Dancer", "Irrelia", "
Irellia", "Ireli", "Relia", "Rellia"], "Ivern": ["
Ivern", "Green Father", "Iver", "Iveren", "Ivern", "
Ivrn", "Daisy"], "Janna": ["Janna", "Storm's Fury", "
```

```
Storms Fury", "Jana", "Jannna", "Jenna", "Jena"], "
Jarvan": ["Jarvan IV", "Jarvan", "J4", "Exemplar of
Demacia", "Jarva", "Jarven", "Jarvan the Fourth", "
Jarvan 4th", "Jarvan the 4th"], "Jax": ["Jax", "
Grandmaster at Arms", "Jaks"], "Jhin": ["Jhin", "The
Virtuoso", "Jin", "Jhinn", "Gin", "Djin", "Djhin"], "
Jinx": ["Jinx", "Loose Cannon", "Jhinx", "Jinks", "
Powder"], "Ksante": ["Ksante", "K'Sante", "Pride of
Nazumah", "Xante", "Kssante"], "Kaisa": ["Kaisa", "Kai
'Sa", "Daughter of the Void", "Kais", "Kaiza"], "
Kalista": ["Kalista", "Spear of Vengeance", "Kallista"
, "Calista", "Callista"], "Karma": ["Karma", "
Enlightened One", "Karm", "Kharma", "Karmma"], "
Karthus": ["Karthus", "Deathsinger", "Kartus", "Kattus
", "Katus", "Khartus", "Kharthus", "Karth"], "Kassadin
": ["Kassadin", "Void Walker", "Kassa", "Kass", "
Kasadin", "Kassadinn", "Kasadin", "Cassadin", "Casadin
"], "Katarina": ["Katarina", "Sinister Blade", "Kata",
 "Katta", "Catarina", "Katerina", "Caterina"], "Kayle"
: ["Kayle", "The Righteous", "Kayl", "Kayel", "Cayle"
], "Kayn": ["Kayn", "Shadow Reaper", "Shaddow Reaper",
 "Kain", "Cayn", "Cane", "Shaddow Assasin", "Rhaast",
"Rhast", "SA"], "Kennen": ["Kennen", "Heart of the
Tempest", "Cennen", "Kenen", "Cenen"], "Khazix": ["
Khazix", "Kha'Zix", "Voidreaver", "Khaz", "Khaziks", "
Khazicks", "Kha'Zics", "Kha'Zicks", "Kha'Ziks"], "
Kindred": ["Kindred", "Eternal Hunters", "Kinndred", "
Kindret", "Kinderd", "Kindered"], "Kled": ["Kled", "
Cantankerous Cavalier", "Cled", "Kkled", "Kledd", "
Cledd"], "Kogmaw": ["Kogmaw", "Kog'Maw", "Mouth of the
 Abyss", "Kog", "Cogmaw", "Coggmaw", "Koggmaw", "Kogg"
], "Leblanc": ["LeBlanc", "Le Blanc", "The Deceiver",
"LeBlanche", "LeBlance", "Le Blance"], "Leesin": ["Lee
 Sin", "Leesin", "Blind Monk", "Lee", "Lesin"], "Leona
": ["Leona", "Radiant Dawn", "Leonna", "Liona", "Leone
"], "Lillia": ["Lillia", "Bashful Bloom", "Lilia", "
Llilia", "Llillia"], "Lissandra": ["Lissandra", "Ice
Witch", "Lisandra", "Liss", "Lizandra"], "Lucian": ["
Lucian", "The Purifier", "Lusian"], "Lulu": ["Lulu", "
Fae Sorceress", "Lullu", "Looloo", "Llullu"], "Lux": [
"Lux", "Lady of Luminosity", "Luxe", "Luxs"], "
Malphite": ["Malphite", "Shard of the Monolith", "
Malph", "Melphite", "Malfight", "Malfite", "Malphight"
```

```
, "Malphit", "Melphit"], "Malzahar": ["Malzahar", "
Prophet of the Void", "Malz", "Malzar", "Malzhar", "
Maltzahar", "Maltz"], "Maokai": ["Maokai", "Twisted
Treant", "Makai", "Mokai", "Maokay", "Mahokai", "
Mahokay"], "Master Yi": ["Master Yi", "Yi", "Wuju
Bladesman", "Master Ji"], "Milio": ["Milio", "Gentle
Flame", "Millio"], "Miss Fortune": ["Miss Fortune", "
The Bounty Hunter", "MF", "Mis Fortunte", "Ms.
Fortunte", "Ms Fortune"], "Mordekaiser": ["Mordekaiser
", "Iron Revenant", "Mord", "Morde", "Mordekaizer", "
Mordekeiser", "Mortekaiser"], "Morgana": ["Morgana", "
The Fallen", "Morg", "Morganna", "Mourgana"], "Naafiri
": ["Naafiri", "Hound of a Hundred", "Nafiri", "
Naffiri", "Nafiiri", "Nafirii", "Naaffiri", "Naafiiri"
], "Nami": ["Nami", "The Tidecaller", "Nammi", "Naami"
, "Nahmi"], "Nasus": ["Nasus", "Curator of the Sands",
 "Nassus", "Nazus"], "Nautilus": ["Nautilus", "Titan
of the Depths", "Naut", "Nauti", "Nauttilus", "
Nautillus"], "Neeko": ["Neeko", "Curious Chameleon", "
Neekoo", "Nekoo", "Neeco"], "Nidalee": ["Nidalee", "
Bestial Huntress", "Niddalee", "Niddale", "Nidallee",
"Nidalle", "Needalee"], "Nilah": ["Nilah", "Joy
Unbound", "Nila", "Nilha", "Nillah", "Neela", "Neelah"
], "Nocturne": ["Nocturne", "Eternal Nightmare", "Noct
", "Noc", "Nocturn", "Nokturn", "Nokturne"], "Nunu": [
"Nunu", "Nunu & Willump", "Willump", "Nunu & Wilump",
"Wilump", "Boy and his Yeti", "Boy and Yeti", "Nunnu",
 "Nunu and Willump", "Nunu and Wilump"], "Olaf": ["
Olaf", "The Berserker", "Ollaf", "Olaff"], "Orianna":
["Orianna", "Lady of Clockwork", "Oriana", "Orianne",
"Oriane", "Ori"], "Ornn": ["Ornn", "Fire below the
Mountain", "Orn"], "Pantheon": ["Pantheon", "
Unbreakable Spear", "Panthenon", "Pant", "Panth", "
Pantenon", "Panteon"], "Poppy": ["Poppy", "Keeper of
the Hammer", "Popy", "Poppey", "Poppeye", "Popeye"], "
Pyke": ["Pyke", "Bloodharbor Ripper", "Pike", "Pyk"],
"Qiyana": ["Qiyana", "Empress of the Elements", "
Quiyana", "Qujana", "Qijana"], "Quinn": ["Quinn", "
Demacia's Wings", "Demacias Wings", "Quin", "Qinn", "
Qin"], "Rakan": ["Rakan", "The Charmer", "Rakkan", "
Rackan"], "Rammus": ["Rammus", "The Armordillo", "
Ramus"], "Reksai": ["Reksai", "Rek'Sai", "Void
Burrower", "Rexai", "Recksai", "Rec'Sai"], "Rell": ["
```

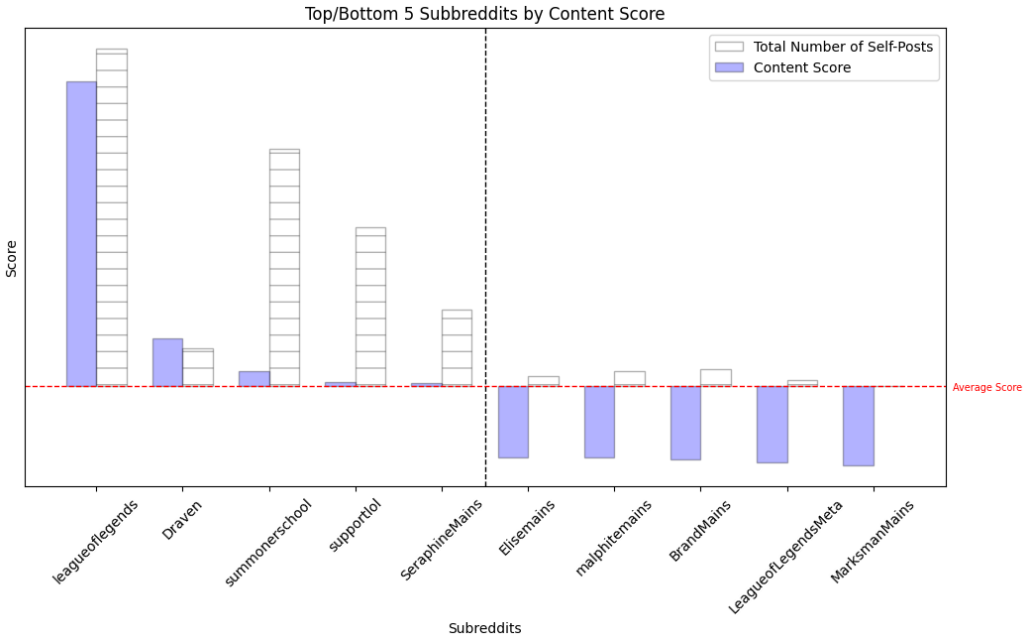
```
Rell", "Iron Maiden", "Rel"], "Renata Glasc": ["Renata
", "Glasc", "Chem-Baroness", "Chem Baroness", "Rennata
", "Renate", "Rennate", "Glask"], "Renekton": ["
Renekton", "Butcher of the Sands", "Renek", "Reneck",
"Reneckton", "Renekten", "Rennekton", "Renneckton"], "
Rengar": ["Rengar", "Pridestalker", "Renngar", "Renger
"], "Riven": ["Riven", "The Exile", "Riwen", "Rivven"
], "Rumble": ["Rumble", "Mechanized Menace", "Rummble"
, "Rumbl"], "Ryze": ["Ryze", "Rune Mage", "Rhyze", "
Ryzhe", "Rize"], "Samira": ["Samira", "Desert Rose", "
Semira", "Sammira", "Semmira"], "Sejuani": ["Sejuani",
 "Fury of the North", "Seguani", "Sechuani", "Sejuanni
", "Seyuani"], "Senna": ["Senna", "The Redeemer", "
Sena"], "Seraphine": ["Seraphine", "Starry-Eyed
Songstress", "Starryeyed Songstress", "Sera", "Seraph"
, "Serafine", "Zeraphine"], "Sett": ["Sett", "The Boss
", "Zett"], "Shaco": ["Shaco", "Demon Jester", "Shacco
", "Shacko", "Shakko", "Shako"], "Shen": ["Shen", "Eye
 of Twilight", "Chen"], "Shyvana": ["Shyvana", "Half-
Dragon", "Half Dragon", "Shyv", "Shivana", "Shyvanna",
 "Shivanna", "Shiv", "Shyva"], "Singed": ["Singed", "
Mad Chemist", "Singe", "Singd", "Singde"], "Sion": ["
Sion", "Undead Juggernaut", "Sione"], "Sivir": ["Sivir
", "Battle Mistress", "Siwir"], "Skarner": ["Skarner",
 "Primordial Sovereign", "Scarner", "Sckarner", "Skarn
"], "Smolder": ["Smolder", "Fiery Fledgling", "
Smoulder", "Smollder"], "Sona": ["Sona", "Maven of the
 Strings"], "Soraka": ["Soraka", "The Starchild", "
Sorraka", "Zoraka", "Zorraka", "Sorakka", "Sorrakka",
"Sora"], "Swain": ["Swain", "Noxian Grand General", "
Swein", "Swayn", "Svain"], "Sylas": ["Sylas", "The
Unshackled", "Silas"], "Syndra": ["Syndra", "Dark
Sovereign", "Sindra", "Synndra"], "Tahm Kench": ["Tahm
", "Kench", "River King", "Tam", "Khench"], "Taliyah":
 ["Taliyah", "Stoneweaver", "Talliyah", "Taliya", "
Thaliya", "Thalliya", "Talliya"], "Talon": ["Talon", "
Blade's Shadow", "Blades Shadow", "Tallon"], "Taric":
["Taric", "Shield of Valoran", "Tarric", "Tarick", "
Tarrick"], "Teemo": ["Teemo", "Swift Scout", "Temo", "
Teemoo", "Temoo", "Timo"], "Thresh": ["Thresh", "Chain
 Warden", "Tresh", "Thres", "Threshe"], "Tristana": ["
Tristana", "Yordle Gunner", "Trist", "Tris", "
Tristanna"], "Trundle": ["Trundle", "Troll King", "
```

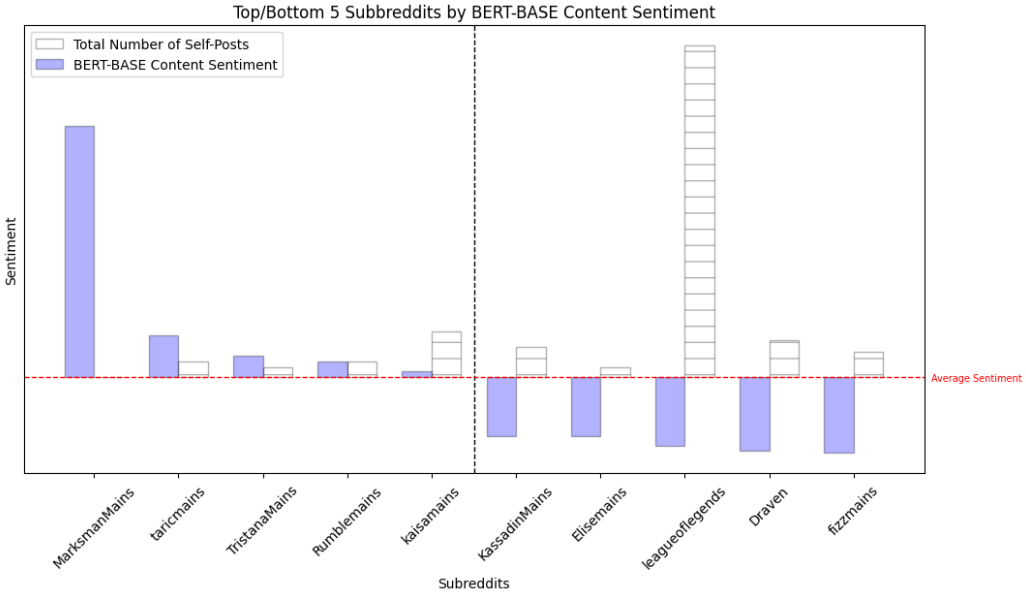
```
Trunndle", "Trundl"], "Tryndamere": ["Tryndamere", "
Barbarian King", "Trynd", "Tryndamer", "Trynda", "
Trindamere", "Trindamer"], "Twisted Fate": ["Twisted
Fate", "Card Master", "Twisted", "TF"], "Twitch": ["
Twitch", "Plague Rat", "Twich"], "Udyr": ["Udyr", "
Spirit Walker", "Udir"], "Urgot": ["Urgot", "
Dreadnought", "Urgott", "Urgod"], "Varus": ["Varus", "
Arrow of Retribution", "Varrus"], "Vayne": ["Vayne", "
Night Hunter", "Vayn", "Wayne"], "Veigar": ["Veigar",
"Tiny Master of Evil", "Veighar", "Viegar"], "Velkoz":
 ["Velkoz", "Vel'Koz", "Eye of the Void", "Vellkoz", "
Velkos"], "Vex": ["Vex", "Gloomist"], "Vi": ["Vi", "
Piltover Enforcer"], "Viego": ["Viego", "Ruined King",
 "Veigo"], "Viktor": ["Viktor", "Machine Herald", "Vik
", "Victor"], "Vladimir": ["Vladimir", "Crimson Reaper
", "Vlad", "Vladd", "Vladdimir", "Vladimmir"], "
Volibear": ["Volibear", "Relentless Storm", "Voli", "
Vollibear", "Volibaer"], "Warwick": ["Warwick", "
Uncaged Wrath of Zaun", "Warwik", "Warwic", "Worwick"
], "Wukong": ["Wukong", "Monkey King", "Wuking", "Kong
"], "Xayah": ["Xayah", "The Rebel", "Xajah", "Xaya", "
Xaja", "Xaiah"], "Xerath": ["Xerath", "Magus Ascendant
", "Xerat"], "Xin Zhao": ["Xin", "Zhao", "Seneschal of
 Demacia", "Zao"], "Yasuo": ["Yasuo", "The Unforgiven"
, "Yas", "Yass"], "Yone": ["Yone", "The Unforgotten",
"Yon", "Ione"], "Yorick": ["Yorick", "Shepherd of
Souls", "Yorrick", "Yoric", "Yorik", "Yorikk", "Jorick
"], "Yuumi": ["Yuumi", "Magical Cat", "Yumi", "Yummi",
 "Yuummi", "Yumii", "Yuumii", "Yumie"], "Zac": ["Zac",
 "Secret Weapon", "Zack"], "Zed": ["Zed", "Master of
Shadows", "Zedd"], "Zeri": ["Zeri", "Spark of Zaun", "
Zerri"], "Ziggs": ["Ziggs", "Hexplosives Expert", "
Zigs"], "Zilean": ["Zilean", "Chronokeeper", "Zilliean
", "Zilian", "Zillean"], "Zoe": ["Zoe", "Aspect of
Twilight", "Zoee", "Zoey"], "Zyra": ["Zyra", "Rise of
the Thorns", "Zira", "Zhyra", "Zyrah"]}
```

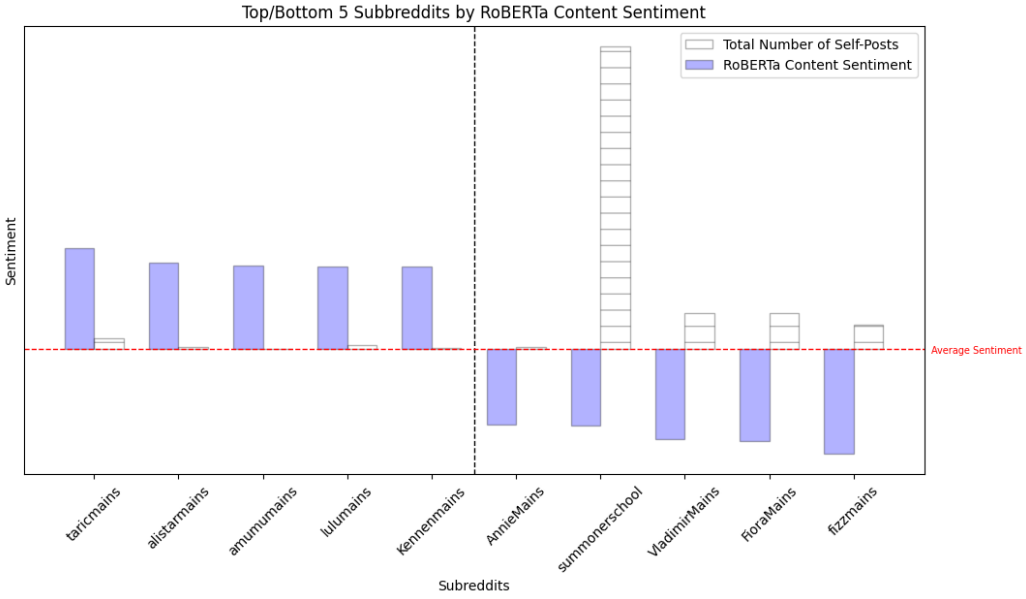
H PyTorch Configuration

PyTorch Build	Stable (2.3.0)		Preview (Nightly)	
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python		C++ / Java	
Compute Platform	CUDA 11.8	CUDA 12.1	~~CUDA 12.4~~	~~ROCm 6.0~~	CPU
Run this Command:	pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu121			

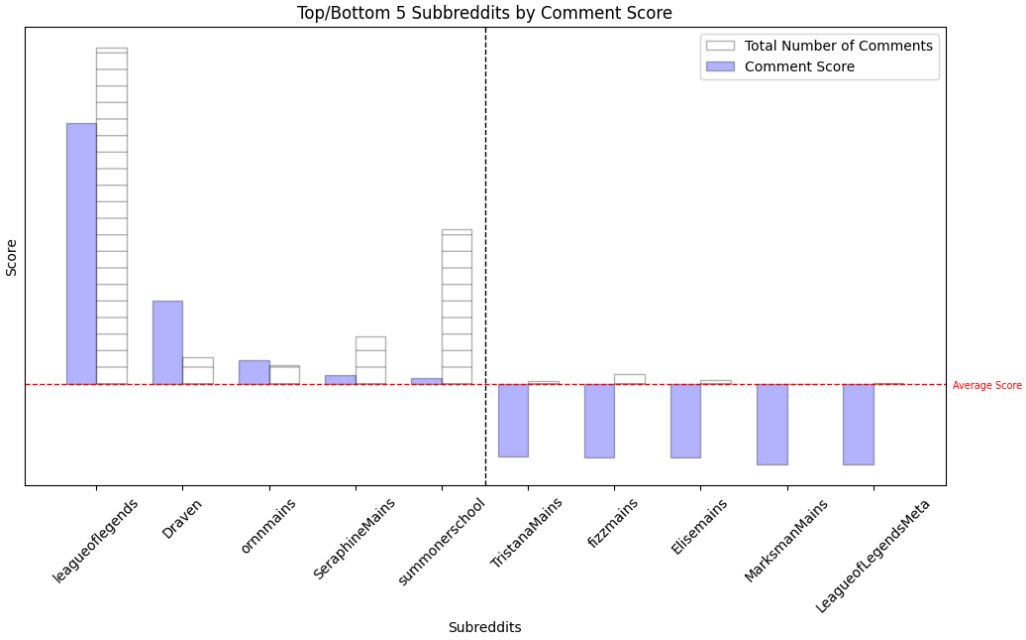
I Top/Bottom Subreddits based on Post Content Metrics

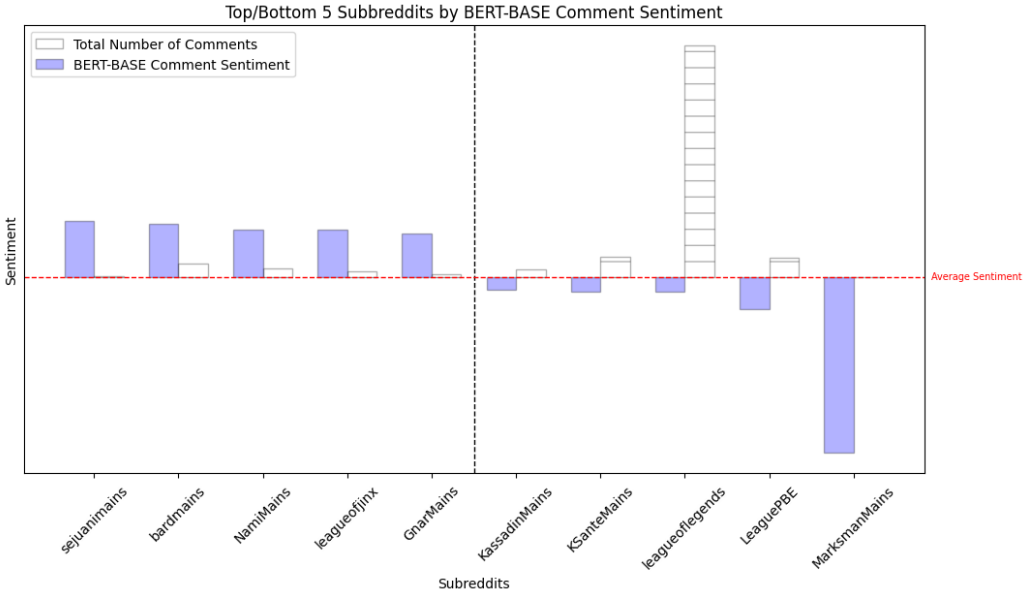


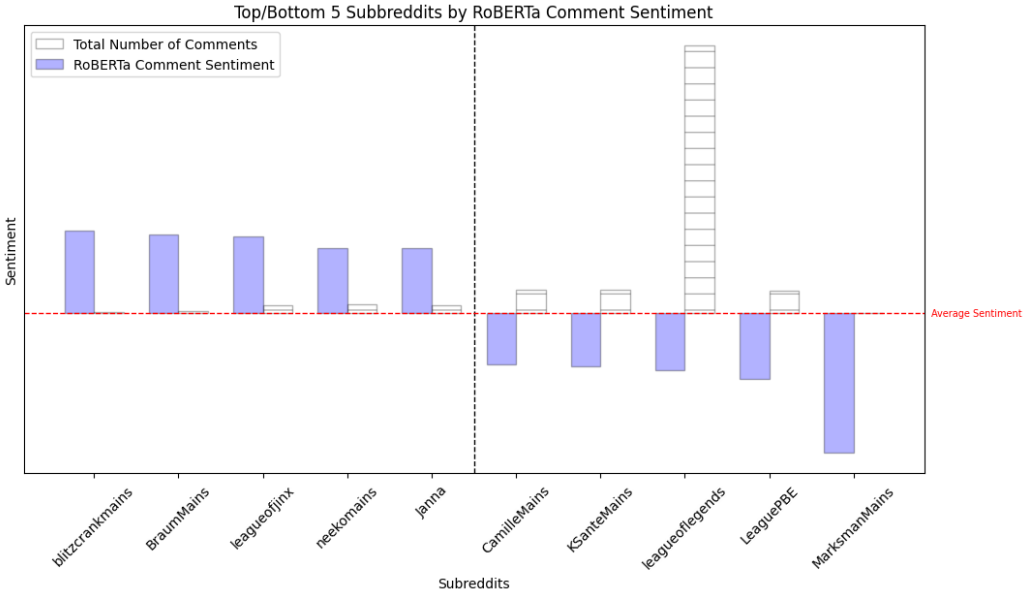

Top/Bottom 5 Subbreddits by BERT-BASE Content Sentiment
Total Number of Self-Posts
BERT-BASE Content Sentiment
Sentiment
Average Sentiment
MarksmanMains
taricmains
TristanaMains
Rumblemains
kaisamains
KassadinMains
Elisemains
leagueoflegends
Draven
fizzmains
Subreddits


Top/Bottom 5 Subbreddits by RoBERTa Content Sentiment
Total Number of Self-Posts
RoBERTa Content Sentiment
Sentiment
Average Sentiment
taricmains
allstarmains
amumumains
lulumains
Kennenmains
AnnieMains
summonerschool
VladimirMains
FioraMains
fizzmains
Subreddits

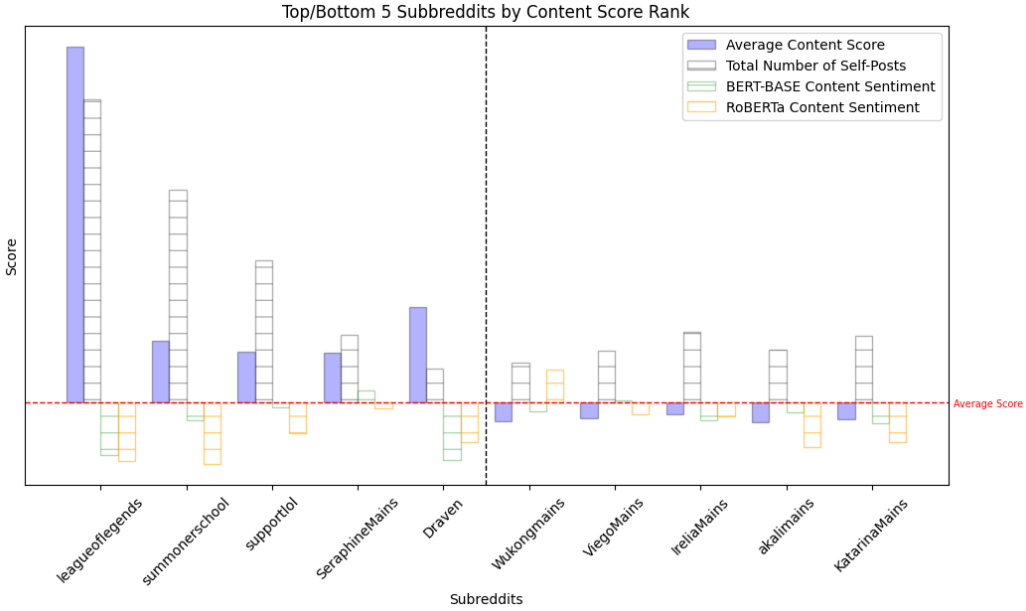
J Top/Bottom Subreddits based on Post Comment Metrics

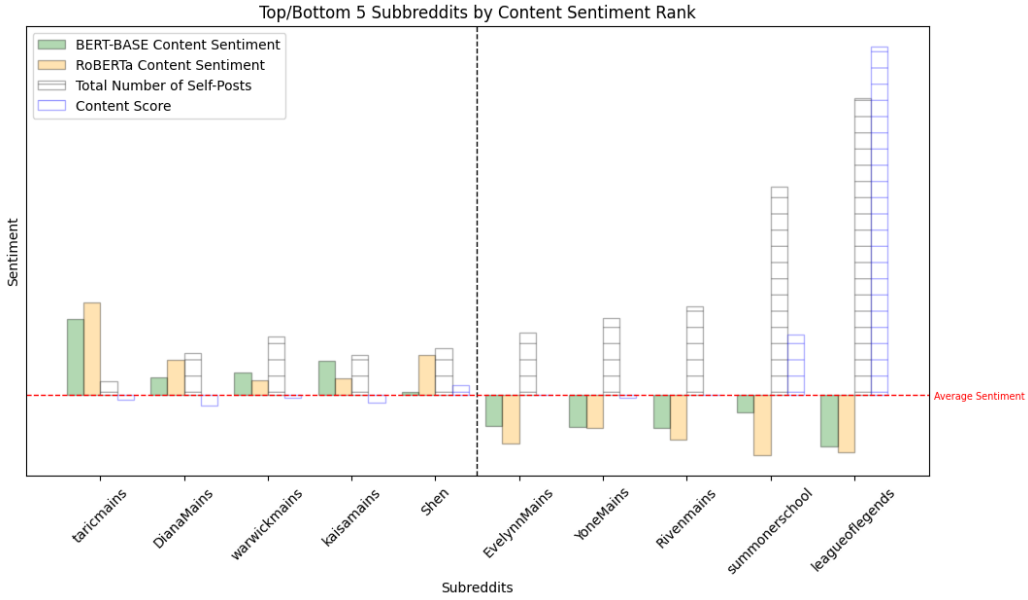




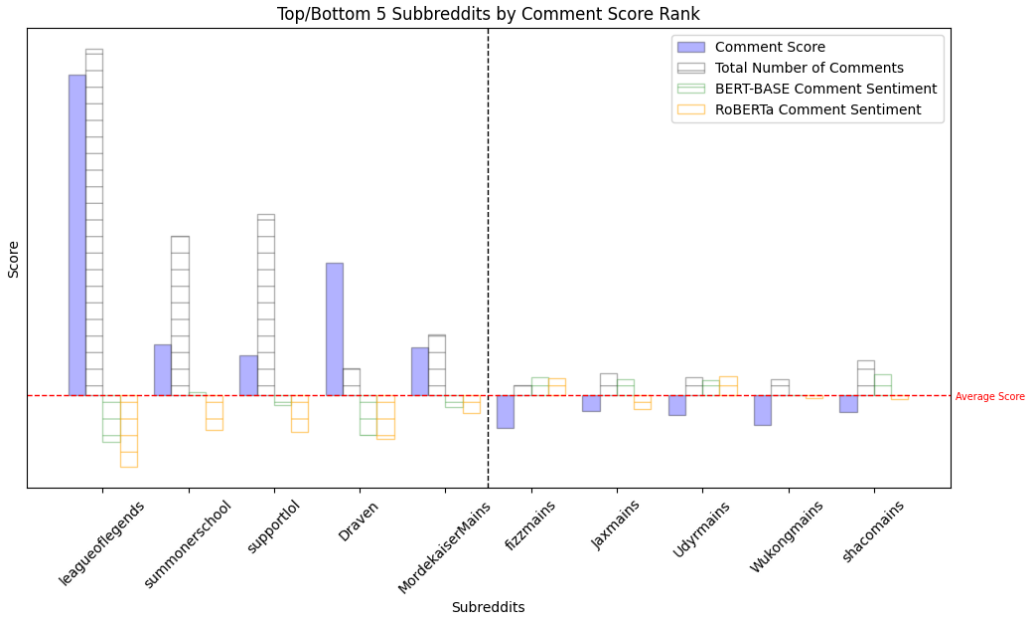


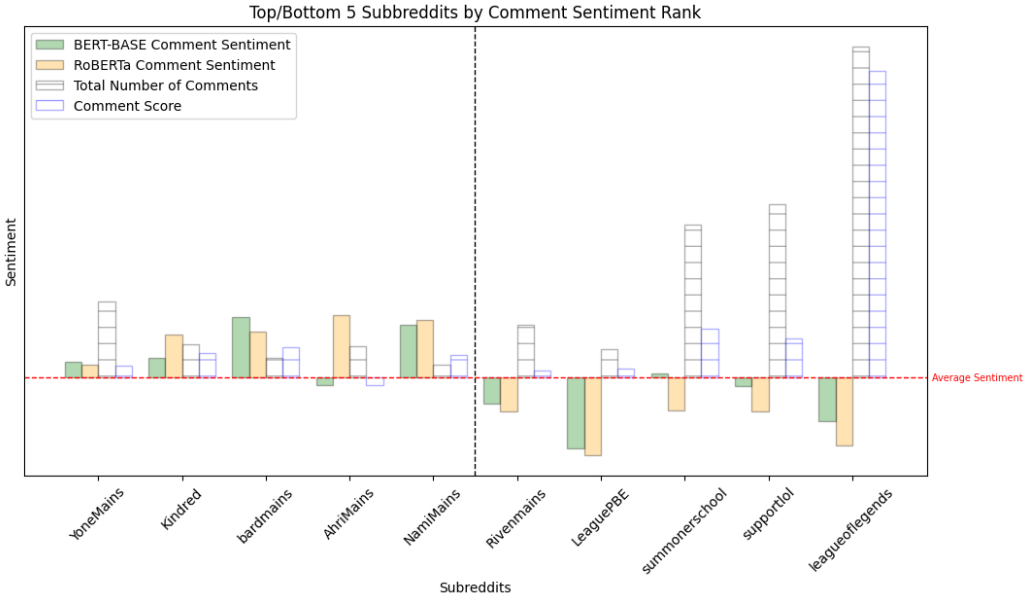
K Top/Bottom Subreddits based on Post Content Ranks



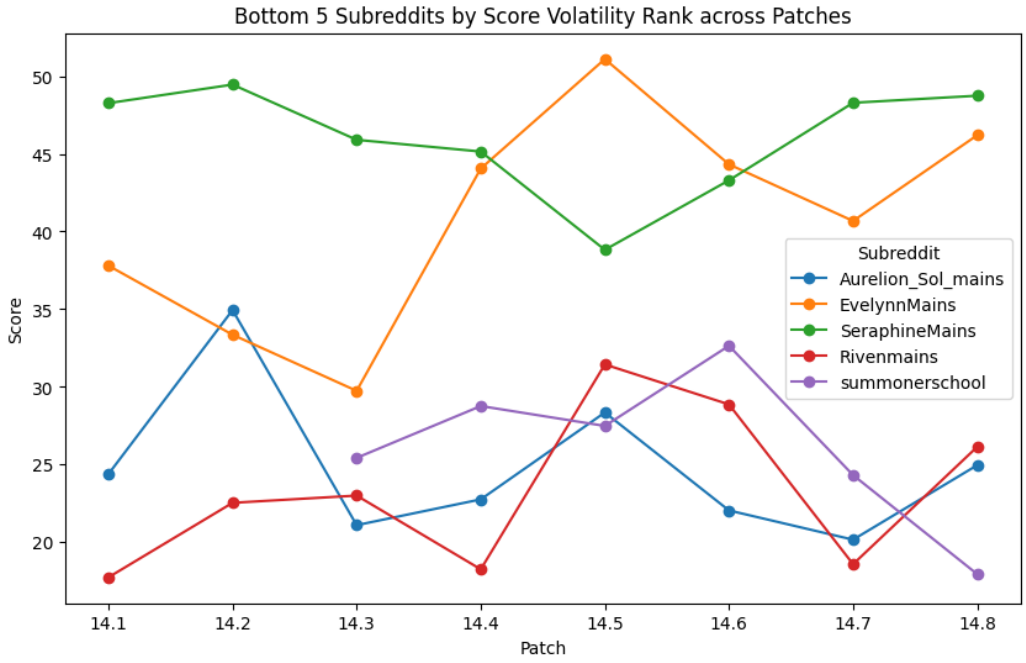


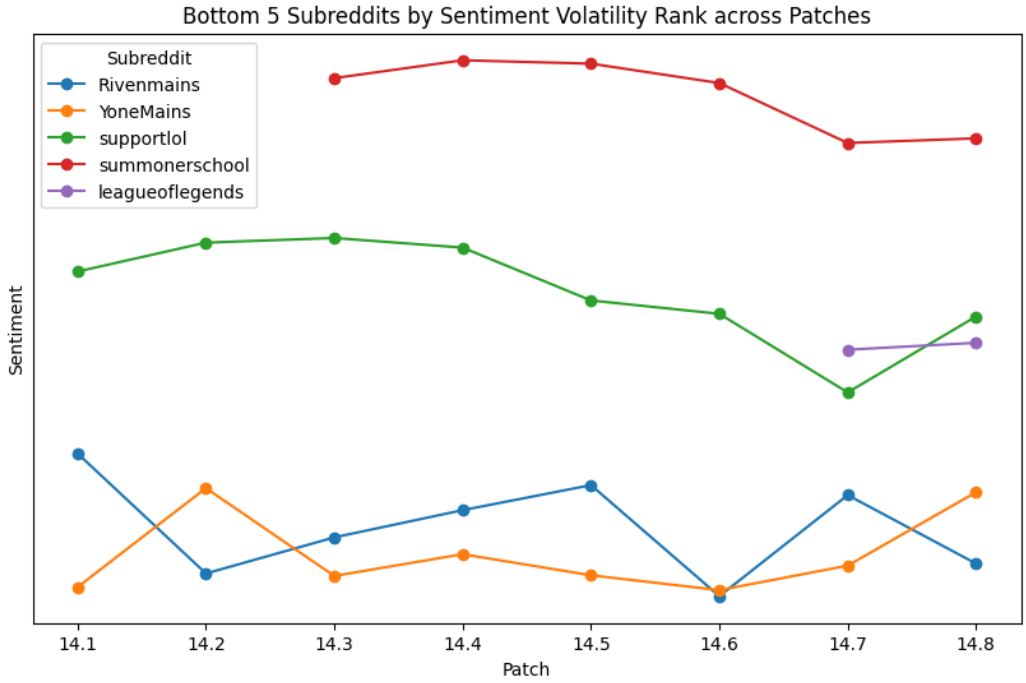
L Top/Bottom Subreddits based on Post Comment Ranks



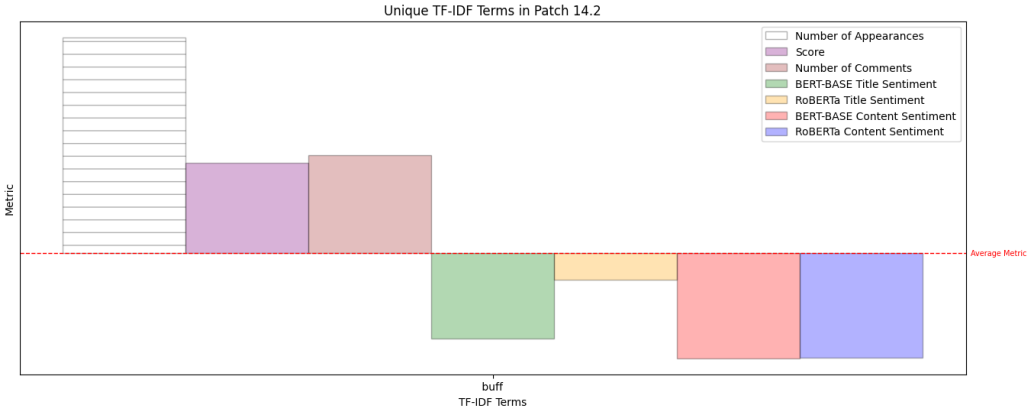


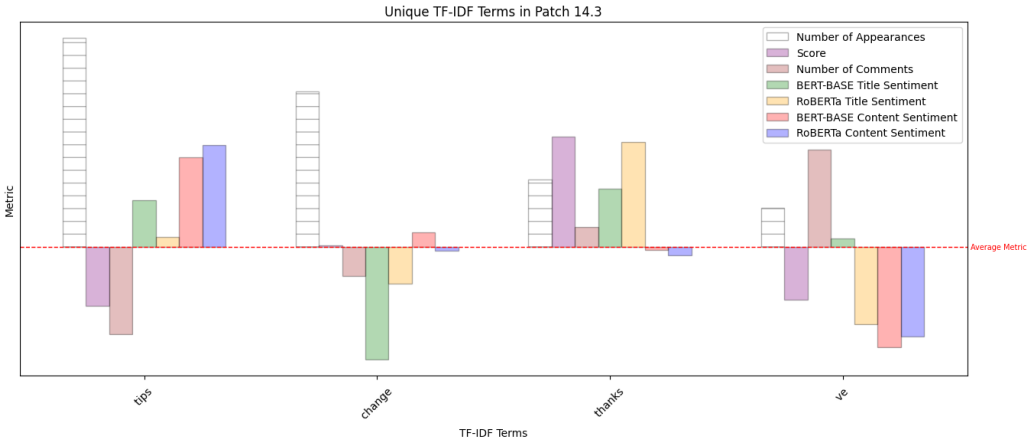
M Bottom Subreddits based on Volatility Ranks

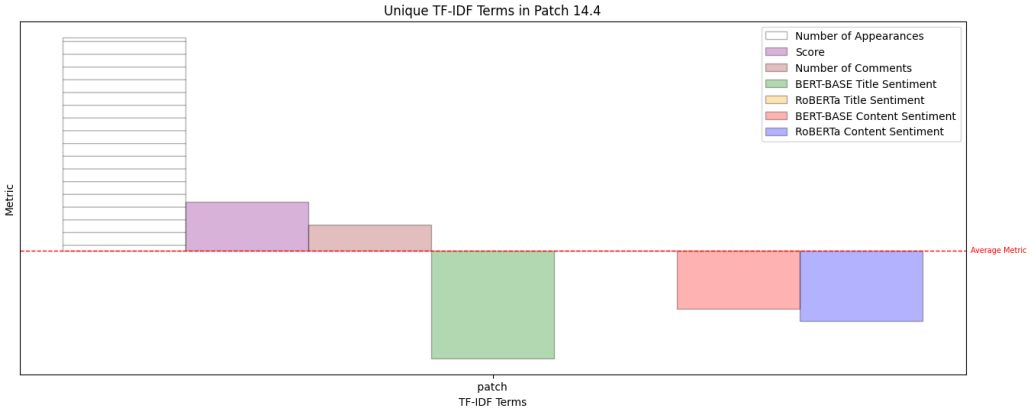


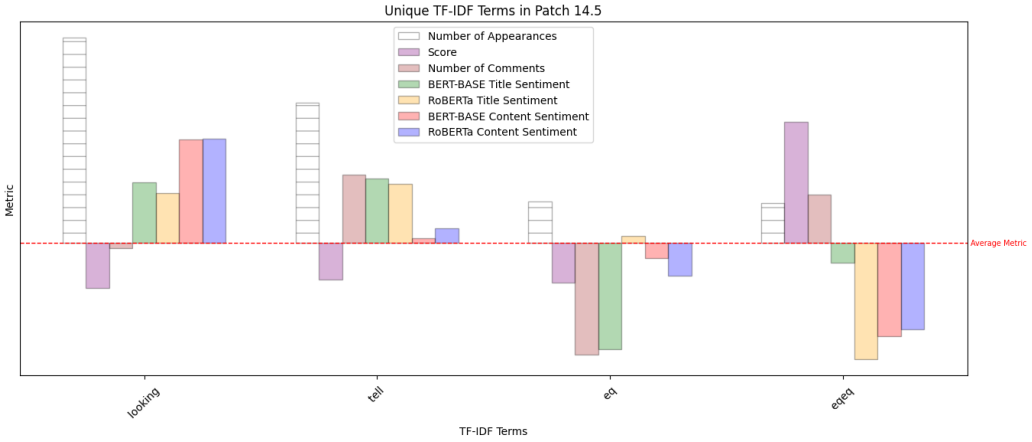


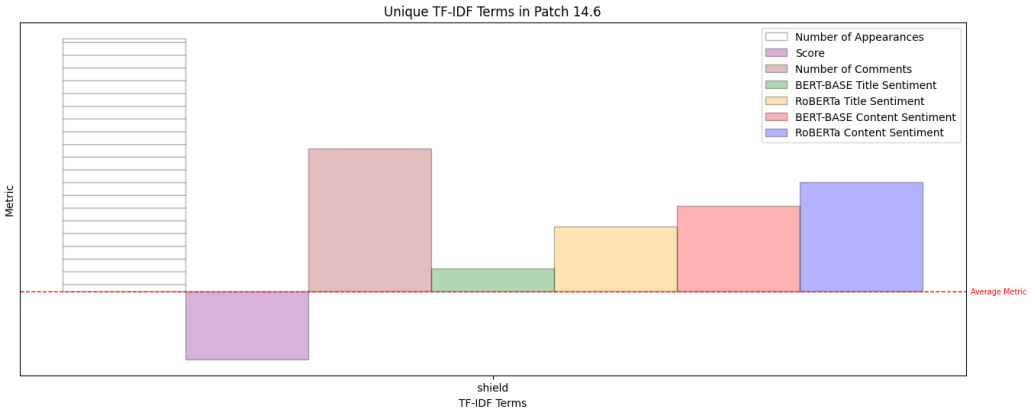
N Unique TF-IDF Term Metrics for Patches 14.2 - 14.8

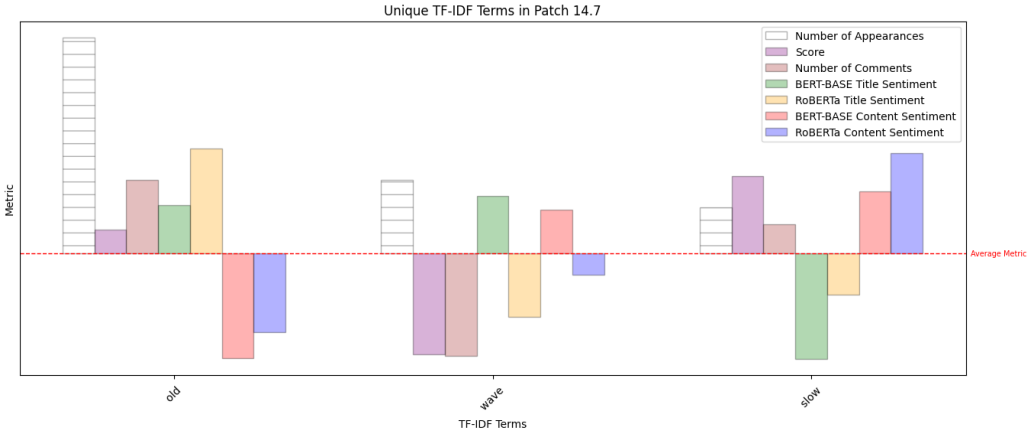


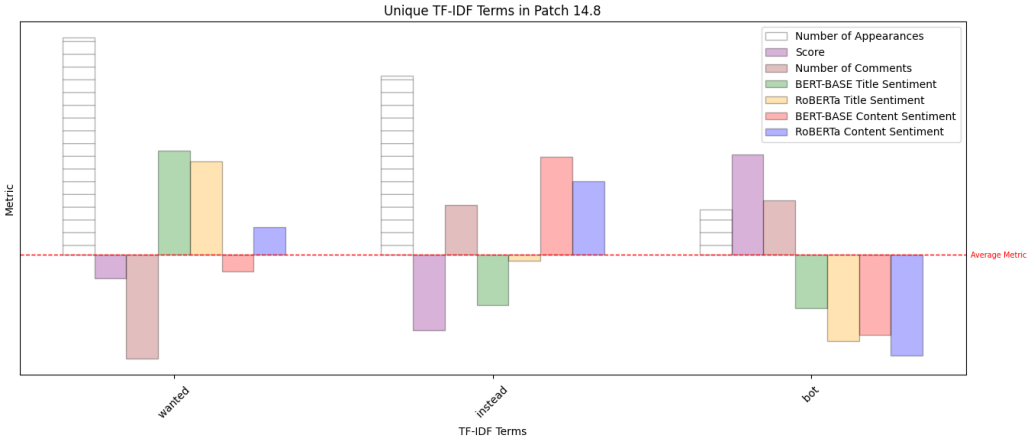

Unique TF-IDF Terms in Patch 14.3
Number of Appearances
Score
Number of Comments
BERT-BASE Title Sentiment
RoBERTa Title Sentiment
BERT-BASE Content Sentiment
RoBERTa Content Sentiment
Metric
Average Metric
tips
change
thanks
ve
TF-IDF Terms


Unique TF-IDF Terms in Patch 14.4
Number of Appearances
Score
Number of Comments
BERT-BASE Title Sentiment
RoBERTa Title Sentiment
BERT-BASE Content Sentiment
RoBERTa Content Sentiment
Metric
Average Metric
patch
TF-IDF Terms

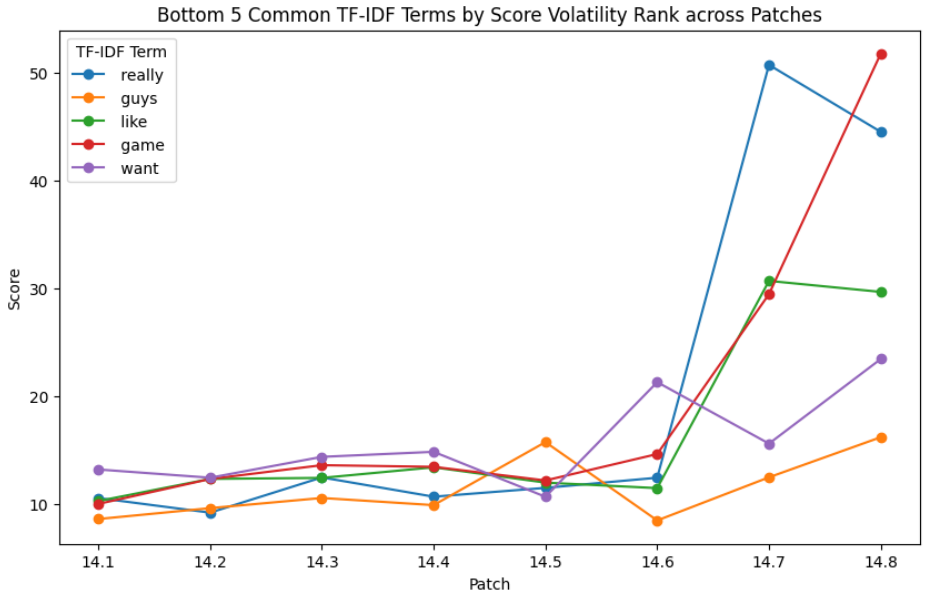

Unique TF-IDF Terms in Patch 14.5
Number of Appearances
Score
Number of Comments
BERT-BASE Title Sentiment
RoBERTa Title Sentiment
BERT-BASE Content Sentiment
RoBERTa Content Sentiment
Metric
Average Metric
looking
tell
eq
eqeq
TF-IDF Terms

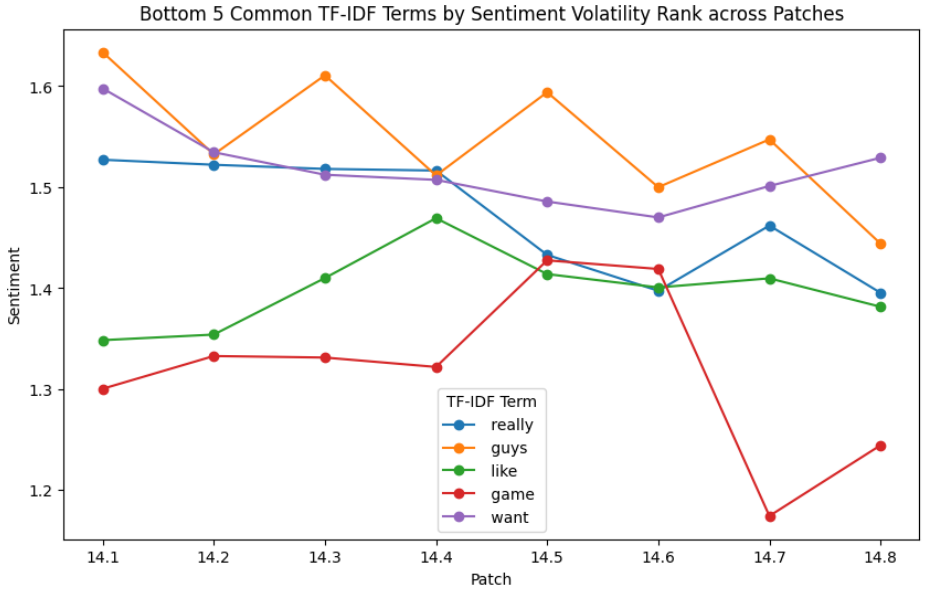

Unique TF-IDF Terms in Patch 14.6
Number of Appearances
Score
Number of Comments
BERT-BASE Title Sentiment
RoBERTa Title Sentiment
BERT-BASE Content Sentiment
RoBERTa Content Sentiment
Metric
Average Metric
shield
TF-IDF Terms


Unique TF-IDF Terms in Patch 14.7
Number of Appearances
Score
Number of Comments
BERT-BASE Title Sentiment
RoBERTa Title Sentiment
BERT-BASE Content Sentiment
RoBERTa Content Sentiment
Metric
Average Metric
old
wave
slow
TF-IDF Terms

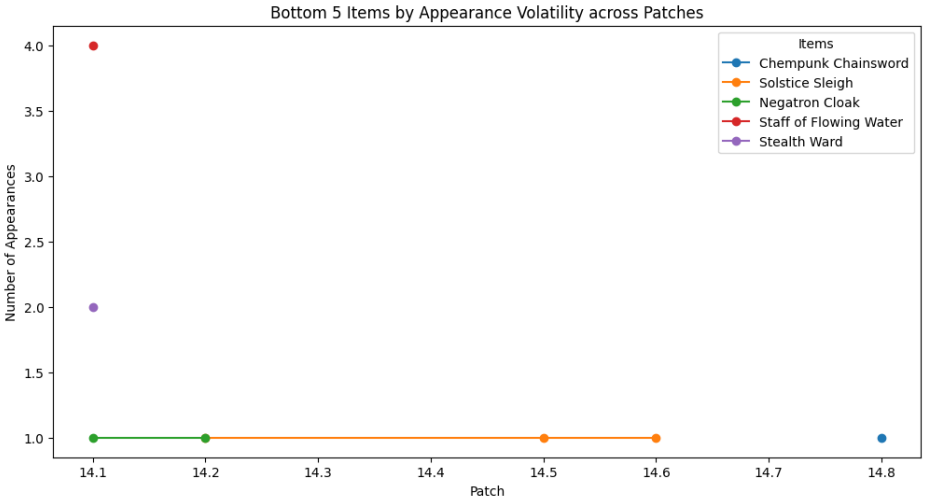

Unique TF-IDF Terms in Patch 14.8
Number of Appearances
Score
Number of Comments
BERT-BASE Title Sentiment
RoBERTa Title Sentiment
BERT-BASE Content Sentiment
RoBERTa Content Sentiment
Metric
Average Metric
wanted
instead
bot
TF-IDF Terms

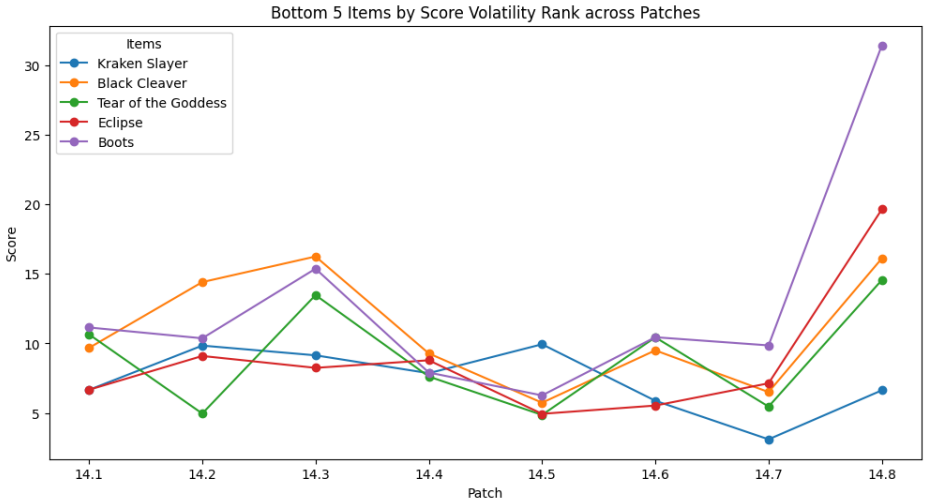
O Bottom 5 Common TF-IDF Terms by Score/Sentiment Volatility Rank

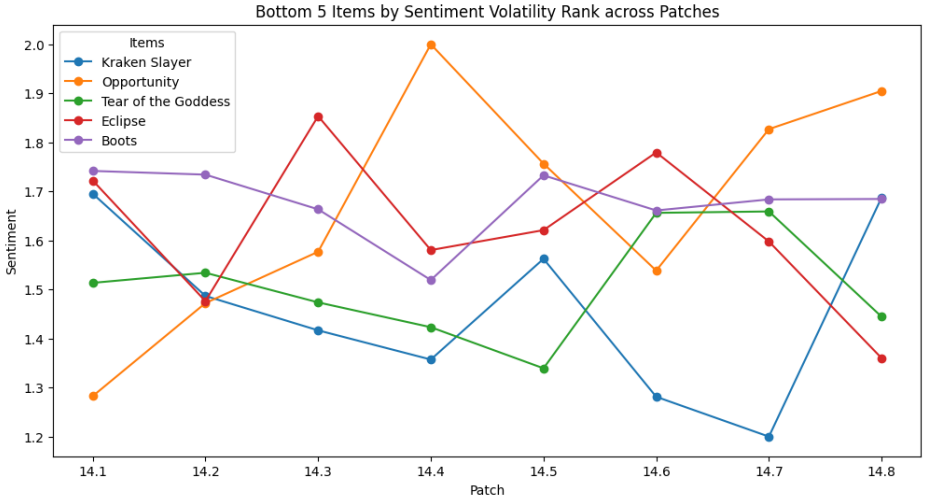




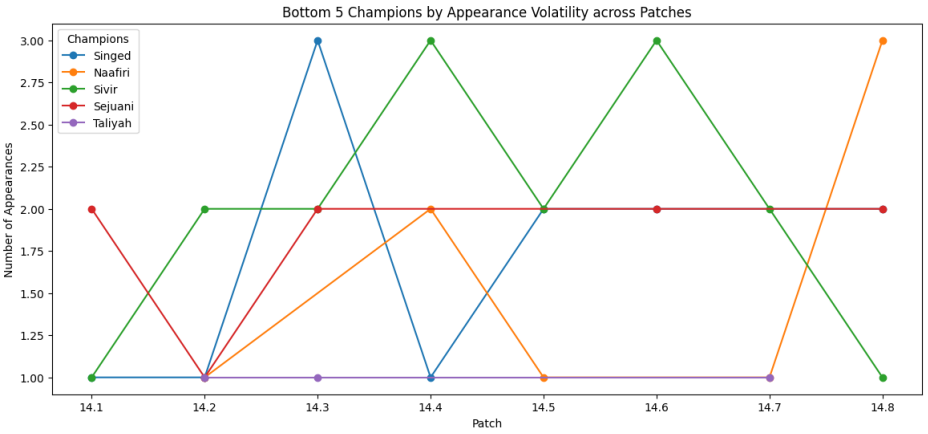
P Bottom 5 Items by Appearance Volatility and Volatility Ranks

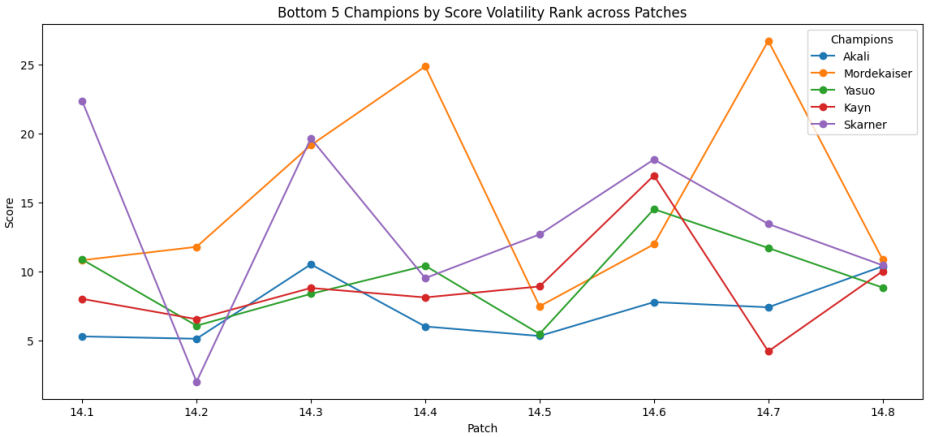


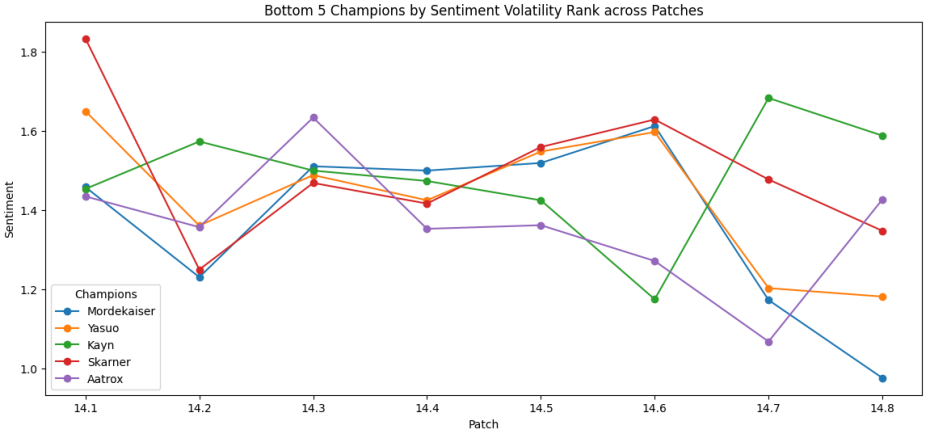




Q Bottom 5 Champions by Appearance Volatility and Volatility Ranks




Bottom 5 Champions by Score Volatility Rank across Patches
Champions
Akali
Mordekaiser
Yasuo
Kayn
Skarner
Score
Patch
14.1
14.2
14.3
14.4
14.5
14.6
14.7
14.8


Bottom 5 Champions by Sentiment Volatility Rank across Patches
Champions
Mordekaiser
Yasuo
Kayn
Skarner
Aatrox
Sentiment
Patch
14.1
14.2
14.3
14.4
14.5
14.6
14.7
14.8