CS:GO Professional Matches and News

Luís Silva, Mariana Costa and Pedro Fernandes*
*Faculty of Engineering, University of Porto (FEUP)
up{201503730, 201604414, 201603846}@fe.up.pt

***Abstract*—This paper concerns the collection and analysis of datasets pertaining to the statistics of professional matches in and news related to Counter-Strike: Global Offensive, a Multiplayer First-Person Shooter which has been cimenting its presence in the Esports scene for the past 8 years. HLTV - one of the most recognizable entities in the CS:GO professional scene and responsible for an extensive database of match statistics and news related to the competitions and its actors - players, teams, coaches, etc. - was used as the source for both datasets. The website was scraped in order to collected said datasets, followed by a cleaning and enrichment processes. From here, the data was analysed in order to understand if the data patterns are semantically expected. Then, the result of this pipeline was fed to an information retrieval system, with the aim of answering domain-specific queries, and the performance of the system was evaluated under different configurations of the indexing and ranking processes. The different configurations of the search engine wielded mean-average precision values between 0.75 and 0.8, considered satisfactory for the task at hands. Solr proved to be an adequate choice for information retrieval tasks; however, unclear and conflicting documentation hinders its potential. Finally, a semantic web approach was studied as an alternative way to structure and reason about the data. Using Protégé, a CS:GO ontology was conceived and tested with a series of SPARQL queries. This approach was then compared to the previous implementation of an information retrieval system, and avenues for future work were defined.**

***Index Terms*—data extraction, data refinement, data analysis, information retrieval, Counter-Strike, Solr, semantic web, Protégé**

I. Introduction

The Counter-Strike series started in 1999 as a Multiplayer First-Person video-game in which two teams - the "Terrorists" and the "Counter-Terrorists" - compete in a series of challenges that involve the "Counter-Terrorists" stopping the "Terrorists" from commiting acts of terror (planting explosives, holding hostages, assassinations, etc.) [1]. Multiple sequels appeared throughout the years, the most recent of which being Counter-Strike: Global Offensive, or CS:GO, as how it will be refered to from this point forward [2]. As of October 2020, CS:GO has amassed a considerable player base, having an average daily peak of around 870,000 players [3], as well as a very active competitive scene, with prize pools reaching the millions [4] and viewership numbers equally as high [5]. This growth has coincided with the rise in popularity of eSports (of which the broadcast of the CS:GO Major Championship in American television is proofc̆iteb9), video game competitions "coordinated by different leagues, ladders and tournaments, and where players customarily belong to teams or other 'sporting' organizations who are sponsored by various business organizations. [6]"

Taking into consideration the aforementioned levels of engagement with the competitive leagues of CS:GO, it is to be expected that matches are heavily documented by the community, with varying degrees of attention to detail. Throughout the years, online platforms such as HLTV [13] and Liquipedia [23] have stood out as information hubs that centralize all matters CS:GO-related. Enumerous statistics are collected from the matches, ranging from player kills (e.g "Natus Vincere vs NiP" [17]) to heatmaps (visual representations of the location of relevant events that take place during a match - e.g. "BLAST Premier Fall Series 2020" [18]), and subsequently analysed in a variety of formats (YouTube [19], platforms that aim to help players improve their performance [20], etc.). Given the amount of data this entails, the need for a search system that allows for querying on multiple criteria is substantial (both HLTV and Liquipedia offer said services, with HLTV usually providing the user with more stastistics). The results of these search tasks allow for the previously mentioned analysis to take place, indicating a teams's potential winning trajectory, a listing of the most commonly strategies used in a given map, how players perform when playing as "Counter-Terrorists" or as "Terrorists", among other various topics.

In addition to match statistics, news regarding the game's competitive scene are also frequently produced. While match performances can provide more factual information on a team's current standing, news can highlight other equally relevant (if more subjective) matters, such as a player's controversies [21], player exchanges between teams [22], and much more. As such, they are necessary to obtain the full picture of CS:GO's professional leagues.

The goal of this paper is to describe the preparation of a search system that allows for the user to obtain information on CS:GO matches and news in an expedited manner. Sections 1 through 3 present the necessary steps to obtain and refine the relevant data, as well as an overview of the datasets, in order to understand the domain and how the information is distributed. In Section 4, the implementation of a search system built upon the documents collected will be discussed. The topics under discussion relate to the selection of tools, to the collections used, to the indexing process and to the retrieval tasks conducted to test the performance of the system. Finally, in Section 5, the implementation of an ontology based upon a subset of the matches dataset will be described. We will present an analysis of existing ontologies, the process for ontology creation and population, the exploration of the ontology through SPARQL queries, the evaluation of the used

tools, a comparison with information retrieval systems, and possible applications.

II. Data Pipeline

In this section, a brief overview on the data sources will be given, as well as a rundown of the main steps behind the cleaning and refinement processes of the two collected datasets.

A. Data Collection

HLTV is an online platform that tracks CS:GO professional matches and offers a way for collaborators to contribute with news pertaining to the video game's competitive scene [13]. It started in 2002 and has since become a hub for Counter-Strike related information. The website counts with nearly 200,000 daily unique viewers [8], and has stood out has one of the most relevant entities in CS:GO journalism. Its importance in the CS:GO scene is recognized by tournaments, who have used their team world rankings for seeding purposes [11], and the game developers themselves, who provide a schedule of professional games in the CS:GO client, using data provided by HLTV [12].

As was previously mentioned, HLTV keeps records of a substantial number of professional CS:GO matches throughout the years, both in minor and major competitions. It is from this database that the dataset on this topic stems from. Since no API is made available by HLTV, the data had to be scraped from the website. The result of this process was a collection of matches held between 2015 and 2020 that was posted on Kaggle (a hub for data scientists and machine learning enthusiasts [7]), and subsequently downloaded in the context of this project [24]. No information on the scraping process was divulged by the author. The dataset was released under a CC BY-NS-SA 4.0 license.

In regards to CS:GO related news, no pre-existing dataset was found; as such, HLTV was scraped in order to obtain all news from 2018 and 2019 using Scrapy, "an open source and collaborative framework" for scraping and web crawling [15]. No information regarding scraped content was found on the source (HLTV); however, it is mentioned that copyright to all content on the website is owned by the platform [25].

B. Data Cleaning and Refinement

Upon a more thorough analysis of the matches dataset, a number of inconsistencies were detected and handled accordingly. For this cleaning process, OpenRefine was used, a tool for data cleaning and wrangling [14]. The tasks revolved mostly around date formating and removal of columns that were deemed irrevelant to the project at hands. In regards to the professional matches, they were additionally filtered to include only entries from 2018 and 2019. As for the news, only the date information was formatted to fit the template adopted when cleaning the matches dataset.

One of the advantages of the match dataset was the inclusion of ID values which allowed the user of said dataset to connect information between the several files it provided (the structure of the data will be detailed in a subsequent section). This opened up the possibility, for example, for different information (from distinct sources) regarding a particular match to be reconciled. However, the news dataset, as it was in its original form, did not provide any way to establish a connection between it and the information on professional matches and its actors. For this to happen, entities needed to be extracted from the news content as a way to establish a bridge between the two domais. From this, a new file which connected news to players and teams would be produced. This posed two problems: which entities should be extracted? And, since there are no limitations as to what a player or team may be called, how will false positives be handled (i.e. a player or team whose name is generic enough to be detected in an abnormal amount of news articles)?

To solve the former, unique player and team names were collected from the dataset on professional CS:GO matches. This list was then fed to spaCy (a NLP tool developed in Python [16]), which proceeded to annotate all ocurrencies of said entities in each news article. The results were then exported to a CSV file.

No systematic solution was found to tackle the problem of false positives. For the data analysis, the entries were reviewed manually; however, that will not be a possibility in the final implemented system.

A graphical explanation of the data pipeline can be found in appendix G.

III. Data Characterization

Having detailed the extraction and refinement processes, the data will now be characterized to a greater extent. An overview of the files that compose the datasets will be provided, including the formats, content structure and number of entries. Then, the conceptual model will be analysed, and finally, some exploratory analysis will be conducted.

The datasets span six CSV files: four pertaining to the match information, and two related to CS:GO-related news. Given the extensive nature of some of the files in terms of columns, the specific file structure will be present in appendix D.

A. Conceptual Model

The conceptual model can be found in appendix H. The focus of the project relies on the player, match team, match and match map. The distinction between a team and a match team stems from the desire to avoid a ternary association between team, player and match. A player plays in a match within a specific team; however, throughout a player's career, they might switch teams, and it is important to preserve a player's contractor at the time of a given match. Match teams aim to accomplish just that; they are a "snapshot" of the composition of a team around the time of a given match. A match team is composed of exactly five players and is associated with a single team. A match has exactly two match teams and, consequently, exactly ten players. Said match is played in between one to five match maps (instances of a map within the context of a given match), has a veto process (in which

teams choose and exclude which maps will be played) and belongs to a tournament. Each map can be played a variable number of rounds, up to a maximum of thirty. Finally, a news article can mention both players and teams.

B. Data Analysis

In order to understand how the data is spread out, and if the data patterns fit the expectations of someone who is familiarized with CS:GO, an analysis of the datasets was in place. The subsequent paragraphs will cover the matches, players and news, respectively.

First, a bar graph representing the number of rounds played in each map during 2018 and 2019 was drawn (Figure 2). A few observations can be made regarding said graph: some maps had significant drops in the number of rounds played between years i.e. Cache and Cobblestone); this can be attributed to the fact that those maps were removed from the professional map roster sometime in 2019. On the other hand, some maps (i.e. Vertigo) started registering rounds only in 2019 for the opposite reason to the one stated above (this map in particular was introduced to the professional roster only in 2019). As a final remark in regards to the maps, some (such as Mirage and Inferno) are considered "safer" picks since most teams train on them more frequently; the number of times these maps were picked substantiates that claim.

In regards to the number of matches played on a monthly basis during 2018 and 2019 (Figure 5), one can conclude that periods of relative high activity levels are contrasted with minor slumps in match numbers (e.g July/August of 2018 vs. September/October of that same year). This can be explained by the fact that teams usually have a season break (which is not set to a particular timeframe nor duration; however they might coincide) [10].

A brief look at the news article's character distribution box plot for 2018 and 2019 (Figure 7) indicates that the length of said articles varies little between the two years. The graph was scaled logarithmically since 75% of all news fall under the 2,500 character mark, while the lengthiest articles have around 42,000 and 39,000 characters (for 2018 and 2019, respectively). Outliers are uncommon and usually represent in-depth analysis and overviews of the annual performance of players and teams. Take the ten articles with the highest character count: they are either highlights of top 20 players that year (e.g "Top 20 players of 2018: dupreeh (5)") or feature articles on players (e.g "From Asia to the world: the story of Bleh"), teams (e.g "A year at the summit: how Astralis wrote history"), and the other CS:GO-related topics e.g "Developing in isolation: The story of Australian CS:GO").

The number of entities and their occurences were also registered in the histogram of Figure 6. During the counting and ordering process, the concerned mentioned in the Data Cleaning and Refinement section was made evident. Figure 6 contains the top 10 entities in news article (during 2018 and 2019). As one can observe, the list is comprised of terms that can easily be misindentified as players or teams (e.g "in, "will", "Will"). Since string normalization should not be applied to the extracted teams and players nouns (e.g "will" and "Will" may represent different entities), and players with generic names are nonetheless valid, a manual exclusion of entries likely to contain a substantial amount of false positives had to be conducted. Upon the conclusion of this process, the revised top 10 entities is summarized in Figure 7.

C. Possible Search Tasks

Platforms such as HLTV and Liquipedia allow for the search of match statistics, teams, players and maps, as well as news (in the case of HLTV). The proposed search system will provide similar services, with the addition of the possibility for searching for entities within the article text. Table 2 summarizes the aforementioned search tasks.

With these tools at the disposal of the user, some examples of relevant queries include:

- Which player performed better on Inferno in November 2019?
- What is the synopsys of the Astralis vs Liquid match?
- How many times have Astralis and Liquid played each other in 2019?
- Is Astralis a CT-sided team?
- Who is the best support player from the USA?
- How many times have Astralis played Vertigo before the StarLadder 2019
  Major?
- Who's the worst player in Astralis?
- How has Team Liquid fared against better opponents in 2019?
- What team has lowest pick win rate in 2019?
- I want to know more about s1mple.

IV. Information Retrieval

Information retrieval can be described as the process through which a user seeks information regarding a particular need, within a particular collection of documents [26]. Let us say a user wants to know more about the reasons behind the recent wave of bans in competitive CS:GO [27]. This information need, as is (i.e. "What are the reasons behind the recent wave of bans in competitive CS:GO?"), does not particularly lend itself to be used directly in search engines. Its structure is too complex and contains terms which are of little discriminative power (i.e. stopwords); it would be beneficial to translate it to a query composed of keywords that describe said need. While some meaning is lost in translation, search engines are optimized to deal with combinations of keywords better than with fully formed sentences. Having determined the query for which to search, the user can now use it in the search system; for the sake of this example, the aforementioned information need will be represented by the query "csgo AND coach AND ban". The system (we will considered it ranked) will then present a list of potential documents ordered by relevance according to a specific ranking function, and it is now up to the user to determine whether or not the retrieved documents answer their information need. In this brief description of the information retrieval process, four

distinct phases can be extracted: indexing, querying, ranking and evaluating.

During the indexing process, the search system builds the vocabulary against which the query will be processed, and determines which fields of a document will be indexed. For example, given a collection whose documents are news articles composed of a title and text, a search system may only index the title.

The interrogation of a search system on behalf of the user is referred to as querying. Queries are composed of terms that aim to synthesize an information need, as well as of operators that can specify the role of a term in a query (e.g. an exclusion operator that tells the system to find documents which do not contain a certain term). When presented with a query, the system will look for documents that match it, commonly ranking them by how close they match the query, as well as by other ranking signals (e.g. number of in/out-links in web search engines). Some search engines, such as Solr, allow the user to increase the relevance of a result should a particular term match a certain condition or be in a particular field (e.g. the user can prioritize documents whose title contains the terms in the query).

Finally, the performance of a information retrieval system is mostly concerned with the relevance of the retrieved results, as well as with the order in which they are retrieved (in a ranked system). While relevance is paramount to the performance of a search system, its evaluation is not as clear cut as its importance. By definition, a relevant document is one which adequately answers a user's information need [26]. However, this information need isn't always evident; in addition, a query usually cannot represent the nuances of a fully fledged information need, and just because a document contains all terms in a query, does not necessarily mean that said document is relevant. Two important concepts related to quality are precision and recall. Precision describes the percentage of retrieved documents which are relevant, and recall describes percentage of relevant documents retrieved. Based on these two concepts, three other evaluation measurements can be listed: the precision at k (i.e. the precision calculated for k retrived documents), the average precision (i.e. the average of precision values calculated whenever a new relevant document is found) and the mean average precision (i.e. "a single measure of quality across recall levels" [26]).

This section describes the implementation and evaluation of a search system built upon a collection of CS:GO professional match statistics and news articles related to its Esports scene. The underlying information retrieval tool is Solr (another popular choice for projects of this kind is Elasticsearch; this option will be briefly discussed and compared with the chosen tool). In addition to detailing the indexing process, a number of information needs will be described and the top results for each will be presented, along with the chosen evaluation measurements: precision at k, average precision and mean average precision. Finally, the results will be analyzed in order to assess the performance of the system.

A. Tool Selection

Given the ubiquity of searching needs across the spectrum of modern-day applications, a search system that can provide end users with approachable, and yet comprehensive, search capabilities over their data collection is a necessity. Database-like queries (e.g. through SQL) are neither intuitive for the end-user (who, typically, lacks the expertise in database management), nor designed for full-text search, an equally omnipresent task in today's Internet-fueled world.

Solr [28] was chosen as the search engine for this project. However, a brief comparison between Solr and Elasticsearch [29] was conducted prior to this decision. Due to time constraints, the systems were analysed somewhat superficially, and only in theory (no objective tests were made). Given the nature of the tasks at hands, a few key features were highlighted as being the most relevant during the selection process of a search system: ability to perform full text search, ease of configuration, ease of installation, intuitiveness of its interface (if there is one), clarity of the documentation and compatibility with most standards used in dataset collection (e.g. JSON, XML and CSV). While features related to topics such as scalability and security are undoubtedly important in applications which are intended to be released to the public, this project is of a smaller scale, and very limited in its user base. As such, metrics related to features of this sort were discarded for this comparison.

Solr and Elasticsearch are, currently, two of the most popular search engines [30]. Both are built on Apache Lucene [31], a text search engine developed in Java. Solr is considerably older and, consequently, better established; despite this, Elasticsearch has risen in popularity over the last few years. While Solr seems to be more focused on advanced information retrieval, Elasticsearch is more geared towards data analytics [32]. This was one of the main reasons behind the decision to adopt Solr.

Thanks to their common origin (Apache Lucene), both systems are capable of full-text search, with an array of search options available to the user (e.g. wildcard, fuzzy, proximity and range). Both Solr and Elasticsearch are heavily documented; however, given Elasticsearch's novelty factor, it is expected that more updated guides are available online. The two systems offer graphical user interfaces; Solr through its Admin UI, and Elasticsearch through Kibana. At first glance, both appear to be similarly intuitive. Initial configurations for the two systems are simple, and, after installation, it is possible to have an instance of both Elasticsearch and Solr up in a few minutes. Finally, Solr is capable of ingesting data from a variety of data sources, including JSON, XML, CSV, PDF and DOCX. On the other hand, Elasticsearch does not have native support for formats other than JSON, something which is easily circumvented through the use of external data shippers (namely, Beats [33]).

B. Collections and Documents

As mentioned previously, two datasets, comprised of 5 files in total, were extracted in the context of this project. The

files contain information on the players' performances during a match, the economy of a match (the money earned by the teams during the rounds), the match results, the maps picked and banned from a match, and news related to the CS:GO professional scene.

A few key decisions were taken prior to the implementation of the search system described in the present section. Firstly, all information regarding the economy of the match was cut. While the statistics contained in said file are of value in a data analytics perspective, the focus of the system is more geared towards information retrieval, and towards full-text search in particular. The economy data was exclusively numerical, and thus of reduced relevance to the task at hands.

Secondly, one of the initial challenges during this stage was the need to convey hierarchical information. A match is played in several maps (which were subject to a selection commonly referred to "picking"), by several players. Since the information on each particular field of interest (i.e. player, match or picks) was spread out through multiple files, the aforementioned structure was lost when importing the files to Solr. In light of this, all files were programmatically grouped together, along with their hierarchical relationships. The result of this process was a single JSON file, comprised of an array of matches, where each match contained general information (e.g. team names, date, the match winner) along with an array of child documents related to the maps selected for the match and the players' statistics.

Finally, the news articles had no direct connection to the matches. That is, even if an article mentioned a particular match, there was no straightforward way to access said it. Upon further inspection of the news document structure in HLTV, it was discovered that articles related to a match contain a section dedicated to a summary of the results, along with the match ID. On the other hand, the ID's of the matches in the dataset corresponded to the ID's used by HLTV to identify them across different sections of their website (i.e. if an article mentioned a particular match ID, the page dedicated to that particular match in the HLTV website and, consequently, the match information in the dataset, had that same ID). Based on these insights, the news were re-scraped to collect the match ID's (if they were present) and automatically assigned to matches if one was mentioned (i.e. an "article" field was added to the match if a corresponding news article was found). News articles which did not directly mention a match were kept as independent documents. In addition, the time frame of the collected articles was extended to a period of 5 years, between March 2015 and March 2020.

Taking this into consideration, the collection now has only 4 different types of documents: news articles, matches, player statistics and match picks. Player statistics and match picks are considered children of a match document; however, they stand as independent documents as well, and can be searched regardless of the match they belong to.

C. Indexing Process

Following the preparatory steps listed in the previous subsection, the data was subsequently imported to Solr. Solr allows for the creation of a "schemaless" collection, in which the user simply imports the data, and the system automatically defines the schema and indexes the data accordingly. This automatically defined schema proved to be inaccurate; for example, a significant amount of fields were handled as if they contained multiple values, when in reality they did not. As such, the initial schema was manually refined to correct these inaccuracies. In addition, some fields were removed from the indexing process, but nonetheless stored. The removed fields were numerical values for which the user is unlikely to either search for or sort by in an information retrieval context (e.g. a player's number of kills).

A number of default field types were used, namely: 'pint' and 'pfloat' (for the numeric fields such as the rating), 'pdate' for dates, and 'text_general' for text fields. The full list of fields, as well as information on whether they were indexed or not, can be found in Table I. Some fields have two types associated with them; however, only one of the two was active at any given time. An explanation of the custom field types mentioned in the table will follow.

TABLE I
INDEXED FIELDS

Document	Fields
Match	team_1 (text_general;csgo_name_general), team_2 (text_general;csgo_name_general), date (pdate), article (text_general;csgo_text_general)
Player	player_name (text_general;csgo_text_general), rating (pfloat)
Picks	N/A
News	title (text_general;csgo_text_general), text (text_general;csgo_text_general), date (pdate)

The issue of hierarchical relationships was then tackled. In order to ensure that Solr could handle nested documents (i.e. documents that were children of others, such as a player who played in a match), two modifications have to be performed: a "root" field must be added, and, if present, the "_nest_path_" field must be removed (since an unlabelled approach was used, in which all child documents are children of a field named "_childDocuments_"). While the official documentation does not offer much information on this topic, some unofficial resources regarding this were found (namely, in Stack Overflow [34]).

The last step of the schema definition pertains to the addition of filters, during the creation of the index and the analysis of the queries. These filters help refine the indexing and querying process by performing common text analysis tasks, such as stemming and stopword filtering. Solr's default field type for what the system infers to be a text field during automatic indexation (in the "schemaless" mode) is referred to as the 'text_general' filter, which removes stop words and upper

case characters. This was used as a starting point to the implementation of a more complex filter, that better suited the needs of the project and the data which it entails. The improved filter, referred to as 'csgo_text_general' , performs the Porter stemmer algorithm for the English language [36], the removal of singular possessives, and synonym expansion (in the query analyser), along with the two other operations mentioned above. The synonym expansion step proved to be particularly useful, since a considerable number of players and teams (as well as other entities related to the game) have different names associated with them. For example, Natus Vincere, a team from the CIS region, is commonly referred to as "Na'Vi".

It should be noted that the synonym expansion could have been done at index time, as opposed to at query time. However, index-time synonym expansion forces the collection to be re-indexed at every change in the synonym file, and phrase queries which contain multi-word synonyms can fail. This is caused by overlapping index terms and synonyms, a phenomenon commonly referred to as "sausagization" [35]. The introduction of the 'SynonymGraphFilter' has allowed for query-time synonym expansion, which was the approach adopted for this project.

In addition to the 'csgo_text_general' filter, the 'csgo_name_general' applies synonym expansion (again, only in the query analyser) and upper case removal to fields which are simply singular units of semantic interest (e.g. team and player names). This information is summarized in Table II.

TABLE II
CUSTOM FIELD TYPES

Field type	Index filters	Query filters
csgo_text_general	Stop, LowerCase, EnglishPossessive, PorterStem	Stop, LowerCase, EnglishPossessive, PorterStem, SynonymGraph
csgo_name_general	LowerCase	LowerCase, SynonymGraph

D. Retrieval Process

In order to test the performance of the search system, as well as the impact of filters and boosts, five information needs were drawn, each queried on three distinct systems: system 1, where the default index was used; system 2, using an improved index; and system 3, using the improved index of system 2 along with boosts specified at query time. Each information need is accompanied by a brief description of its aim and any relevant characteristics of the query, along with the description of the search parameters and top ten results for the query (including precision at ten and average precision values). The query parser used will be eDisMax.

1) Matches played between Astralis and Natus Vincere (Na'Vi) during BLAST Tournaments: With this information need, we wish to know more about matches from the BLAST tournaments in which Astralis played against Natus Vincere. The use of the boolean operator "AND" mandates the co-existence of the three terms in the search results. The search will be performed on the article, team_1 and team_2 fields of match documents. One of the distinctive features of this query is the presence of a term subject to synonym expansion; for System 1, which does not employ any filter of the sort, the term "navi" was replaced by "natus vincere" (otherwise, no results would have been registered). For System 3, a bigger boost was applied to the team_1 and team_2 fields, as the presence of the keywords in said fields guarantees that the matches were between the two teams, whereas if they occur in the article, there is a possibility that the teams just happen to be mentioned, even if they did not participate. This rationale applies to all queries which apply this sort of boosting from here on out. The results obtained for this information need can be found in Table III.

TABLE III
Q1 PARAMETERS AND RESULTS

Systems 1 and 2	q=astralis AND "natus vincere" AND blast
System 3	q=astralis AND "natus vincere" AND blast, qf=article team_1^5 team_2^5

	1	2	3	4	5	6	7	8	9	10	P@10	AP
System 1	R	R	R	R	N	R	N	R	N	N	0.6	0.931
System 2	R	R	R	R	N	R	N	R	N	N	0.6	0.931
System 3	R	R	R	R	R	R	N	N	N	N	0.6	1

2) Grand finals played by Astralis: With this information need, we want to find matches belonging to the grand finals of tournaments in which Astralis took part. The terms "season" and "edition" were added given the fact that a considerable number of articles tend to mention them when announcing the winner of a tournament. "Grand" was added to differentiate from other finals that might be reported (e.g. semi-finals). The search will be performed on the article, team_1 and team_2 fields of match documents. The results obtained for this information need can be found in Table IV.

TABLE IV
Q2 PARAMETERS AND RESULTS

Systems 1 and 2	q=+astralis final grand ("win edition"~10) champions crown title,
System 3	q=+astralis final grand ("win edition"~10)^10 champions crown title, qf=article^5 team_1^10 team_2^10

	1	2	3	4	5	6	7	8	9	10	P@10	AP
System 1	R	N	R	R	N	R	N	R	N	R	0.6	0.718
System 2	R	N	R	R	N	R	N	R	N	R	0.6	0.718
System 3	R	R	R	R	N	N	R	N	R	R	0.7	0.869

3) Transfers into/out of Cloud9 during 2018: With this information need, we want to find out more on Cloud9's player movements in 2018. Player transactions are common in CS:GO's professional scene; one can think of these transactions as similar to ones occurring in more traditional sports such as soccer. We're looking for transactions in both directions (into and out of the team), and as such, the query reflects this through the inclusion of both "exit" and the remaining terms. The term "Cloud9" must be included in the fields over which the search is being conducted, namely the text and title of independent news articles (i.e. news articles which are not associated with any match). For System 3, it is more valuable for terms to appear in the title of an article, and the boosts encompass this. The results obtained for this information need can be found in Table V.

TABLE V
Q3 PARAMETERS AND RESULTS

Systems 1 and 2	q=+cloud9 transfer sign add join confirm exit, fq=date:[2018-01-01T00:00:00Z TO 2018-12-31T00:00:00Z]
System 3	q=+cloud9 transfer sign add join confirm exit, qf=title^10 text^5, fq=date:[2018-01-01T00:00:00Z TO 2018-12-31T00:00:00Z]

	1	2	3	4	5	6	7	8	9	10	P@10	AP
System 1	R	R	R	N	N	N	R	R	N	R	0.6	0.799
System 2	R	R	R	N	N	N	R	R	R	N	0.6	0.811
System 3	R	R	R	R	R	R	R	R	R	N	0.9	1

4) Matches where FURIA were aggressive: FURIA is a brazilian team that rose to the top of professional Counter-Strike in 2018. They have a trademark style of play, defined as "relentless aggression": their strategies rely on engaging in gun-fights with their opponents in unexpected situations, and they have managed to make this unpredictable style work quite well. Therefore, we want to explore how this notion translates to the news coverage, by finding matches where such aggressiveness was highlighted. To do so, we query for articles were both FURIA and variations of the word "aggression" are found. The results obtained for this information need can be found in Table VI.

TABLE VI
Q4 PARAMETERS AND RESULTS

Systems 1 and 2	q=(furia AND (aggressive OR aggression OR aggressiveness))
System 3	q=(furia AND (aggressive OR aggression OR aggressiveness)), qf=team_1^10 team_2^10 article

	1	2	3	4	5	6	7	8	9	10	P@10	AP
System 1	R	R	N	R	R	R	N	R	R	R	0.8	0.809
System 2	N	R	R	R	N	R	R	R	R	N	0.7	0.613
System 3	N	R	R	R	R	N	R	R	R	N	0.7	0.633

5) Matches won by Natus Vincere (Na'Vi) in 2019: Natus Vincere is at this time the 3rd best team in the world, and holds in their ranks one of the all time Counter-Strike greats: Aleksandr "s1mple" Kostyliev. We want to retrieve matches won by Na'Vi in the year 2019. Similarly to previous queries, we explore Solr's synonyms filter, and in the third system employ a boost function in order to make recent matches more relavant. The results obtained for this information need can be found in Table VII.

TABLE VII
Q5 PARAMETERS AND RESULTS

Systems 1 and 2	q=natus vincere AND (win OR victory), qf=article team_1^10 team_2^10,
System 3	q=natus vincere AND (win OR victory), qf=article team_1^10 team_2^10, bf=recip(ms(NOW,date),3.16e-11,1,1)

	1	2	3	4	5	6	7	8	9	10	P@10	AP
System 1	N	R	R	R	N	N	N	R	N	R	0.5	0.489
System 2	N	R	R	R	R	N	R	N	N	R	0.6	0.588
System 3	R	N	N	R	N	R	N	N	R	N	0.4	0.488

E. Tool Evaluation

While five information needs are not enough to extensively evaluate the performance of a search engine, the results provided by the retrieval process described above shed a light onto how different configurations of the same search engine might affect the outcome of a search task. Furthermore, they highlight the importance of an appropriate query, as this was often the most determining factor in a successful search.

The mean average precision (MAP) for the three systems is similar, ranging from 0.75 to 0.80. The precision values between systems for a particular information need are usually smaller and fluctuate more, registering values between 0.4 and 0.9. The first three information needs are in line with what was expected: the average precision increased with each improvement made to the system, culminating in the performance of System 3. On the other hand, the results of the final two search tasks were somewhat unexpected: for the fourth search task (matches where FURIA's aggressiveness was highlighted), the perfomance of Systems 2 and 3 was lower than the performance of System 1; for the fifth task, only System 3 performed worse than System 1. These results can be seen in Table VIII.

TABLE VIII
MEAN AVERAGE PRECISION

System 1	System 2	System 3
0.749	0.732	0.798

In regards to the fourth search task, one possible reason considered initially for the dip in performance was improper

stemming. Given that, with Systems 2 and 3, custom field types which applied stemming algorithms were used, the query was reduced to include only one term related to aggressiveness (as opposed to the multiple terms used in the first system to increase the chances of finding related documents). However, tests (in which all related words were searched for individually to understand the impact stemming might have) later revealed that the stemming filter was working correctly. As such, the cause behind the surprising poor performance of the improved systems remains unknown. As for the fifth and final task, the boost attributed to recent matches may have left relevant documents out of the top ten results given their age.

Finally, Solr proved to be a versatile search server, with a multitude of customizable settings and search options that allow the user to perform complex queries. However, the integration of nested documents was overly complicated, with unclear documentation being one of the main contributors to this.

While not related to the perfomance of Solr, it should be noted that this collection of documents proved difficult to work with from an information retrieval context, as only one source of unstructured text was available. Additionally, it seems that the potential of the documents is seized much more clearly when adopting a data analysis perspective, and less so in the context of the project at hands.

V. Semantic Web

Having analysed the data at our disposal through the lens of an information retrieval system, we will now explore an alternative way to structure and reason about it: adopting a semantic web approach. This exploration is particularly relevant for the domain in question because, as will be later discussed, not much work has been done in the way of exploring it using this approach.

According to Berners-Lee et al. [37], the goal of semantic web is to turn the Web into a more machine-friendly place, which can allow computers to perform complex tasks without the need for much human intervention. It is seen as an extension of the Web, and not a replacement; something to give it more structure and to help convey semantic meaning to a computer agent as efficiently as it does to a human one. It requires "structured collections of information and sets of inference rules" used to "conduct automated reasoning". Its challenge is then to accomodate for data and rules on how to work and interpret with said data, rules which may already exist in a knowledge-representation system.

The collections of information it relies upon are called ontologies; in the computer science field, an ontology is "a document or file that formally defines the relations among terms." Ontologies most commonly seen on the Web are composed of a taxonomy (i.e. classes of objects and relations among them) and a set of inference rules. Some of the advantages of ontologies applied to the web include the improvement of web searches (by removing ambiguities) and the added ability to tackle complex questions whose answer require the exploration of multiple web pages.

Returning to our domain, in the subsequent sections we will describe the process through which our ontology was created. Firstly, we will discuss some existing ontologies which bear some resemblance to the topic at hands, but which ultimately cannot define it as we would wish them to (leading us to construct most of our classes and relations). Then, our ontology will be described in depth, going over the classes, data and object properties, as well as restrictions. In order to explore our dataset in a more realistic setting, a few SPARQL queries were used to exemplify possible data needs a potential user of an ontology of this sort could have. Finally, we will conclude this section with a brief evaluation of the tools used throughout the process, accompanied by a discussion of the implemented ontology, its shortcomings, and possible future directions.

A. Existing Ontologies

While we couldn't find any ontology directly related to Counter Strike, we can still establish a connection to more traditional sports in regards to entities such as players, teams and competitions. Having this in mind, BBC's sport ontology seems like an appropriate match [38].

It is "a simple lightweight ontology for publishing data about competitive sports events". It originated in 2010, when BCC realized that conventional content management systems impose limitations on the flexibility of the web pages they construct. This limits the richness of the experience they can offer visitors to their site. As such, they collaborated with Epimorphics to develop this ontology [39]. Since its introduction, this ontology was used by BBC during the coverage of the 2012 Summer Olympics and the 2014 Football World Cup.

This ontology is primarily used to cover competitions, providing terms and properties to convey information such as events, players, teams, awards, etc. These broader concepts fit the Counter-Strike domain well, and they appear to be a solid base upon which to build a more specific ontology geared towards the specificities of the game. We also explored the video game ontology [42]; however, its focus lies on the implementation of video game mechanics (such as leaderboard, achievement, item), and not so much in the competitive panorama of online games such as CS:GO.

While the BBC's ontology does provide some fundamental concepts, we have opted to design our ontology from scratch. On the one hand, this approach somewhat contradicts the principle of reusibility inherent to the semantic web; we acknowledge this fact and conclude that the integration of other ontologies could be an avenue for future work. However, we also propose that this kind of ground-up tactic allows us to become more familiarised with the process of ontology creation.

B. Domain Revision

The domain of CS:GO was extensively described in previous sections; nonetheless, we believe it necessary to briefly describe the changes and simplifications made for the creation

of our ontology. In our previous exploration of information retrieval task, there was a heavy focus on news pertaining to CS:GO professional competitions. This was necessary given the lack of unstructured text information in our statistical dataset, which would result in lackluster experiments in the evaluation of the information retrieval tool used. Since the focus of this section does not rely on this type of text-heavy information, we have opted to focus more on the main concepts of the competitive scene.

Some simplifications were made, much in line with what we did in the previous section; the information on map performance was removed and only match performance was kept. In addition, the economy information was not included either, as adapting the files to a more comprehensible format that could allow us to construct a concise yet complete taxonomy of this sub-topic would be cumbersome and would not bring anything new.

As main concepts, we have maps and match maps (instances of maps played in a particular match), players and matchplayers (following the same reasoning, matchPlayers are instances of players who played in a given match for a given team), teams, picks and events. Events, in particular, is a section of the dataset to which we had not paid much attention thus far.

Despite the simplifcations made, we believe the domain retains enough information to adequately represent the subject at hands. It keeps its mains characteristics while being manageable, and it has enough information to allow for the study of the inner workings of building, populating and exploring an ontology, as well the understanding of its usefulness.

C. Ontology Creation

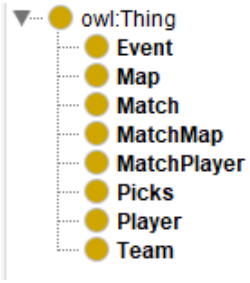

Fig. 1. Ontology classes

The implemented classes can be seen in Figure 1. They aim to convey all essential information previously described in the conceptual models, while allowing for some flexibility in terms of what can be simplified and what should remain as is. All classes are commented, and most concepts have already been detailed in previous sections. As such, we will refrain from explicitly describing them here. The "events" class was planned to use the BBC Sport Ontology's competition concepts as sub-classes. However, our dataset does not allow for such distinction, and we have opted to consider all events mentioned as independent. One possible improvement to this would be to use NLP tools that would extract the competition, as well as the edition/year of occurrence, which could then be used to define a more explicit web of concepts regarding this topic (e.g. an event could be an overarching competition such as IEM Katowice which would then be described by data properties such as the year, as opposed to considering these elements as a single property).

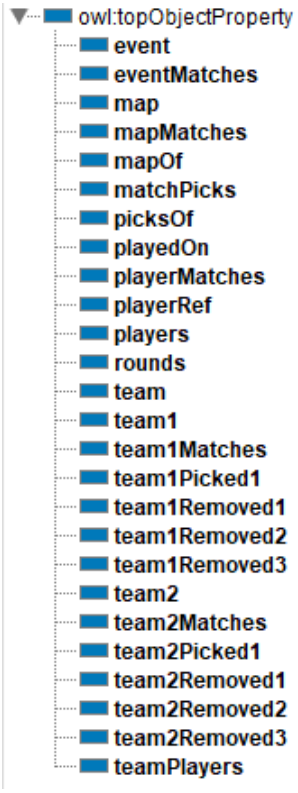

Fig. 2. Ontology object properties

In regards to the object properties (defined in Figure 2), similarly to the implemented classes, some have already been explained and/or are accompanied by comments. Nonetheless, one topic should be explained more thoroughly: the properties pertaining to the "picking" process, i.e. those with the prefixes "teamXRemoved" and "teamXPicked". Before a match starts, both teams exclude and pick the maps which will or will not be played. The order in which the selection/exclusion is made, along with the team to which it belongs, are relevant from an analytical perspective. For instance, one of the SPARQL queries we implemented intends to find which maps a team bans first the most (which, in essence, means the team is not comfortable playing on such maps). In addition, the number of picks has an upper bound (i.e. at most, 6 maps are removed and 2 are picked). For these reasons, we have opted to explicitly describe each step in the process through the use of the aforementioned properties. One alternative approach would be to include "picked" and "removed" classes, which would then be described by properties such as the team to which it belongs, its order in the selection process and the map which was selected/excluded. This approach would require an extensive overhaul of the dataset, which we deemed

unnecessary.

All object properties were declared as functional, and the inverse of most was defined (Table IX presents these property pairs), with the exception of the classes pertaining to the picking process. We believe that since those classes are closely tied to our dataset in particular and not necessarily generic, we have opted to simplify the process and exclude said properties.

TABLE IX
PROPERTY/INVERSE PAIRS

Property	Inverse
event	eventMatches
map	mapMatches
team1	team1Matches
team2	team2Matches
playerRef	playerMatches
team	teamPlayers
players	playerOnd
matchPicks	picksOf
rounds	mapOf

Finally, the data properties present in Figure 3 include some of the concepts more closely tied to the mechanics of the game (as with the classes and object properties, they are accompanied by relevant comments which aim to succinctly clarify them). All except but one data property (name) were declared as functional.

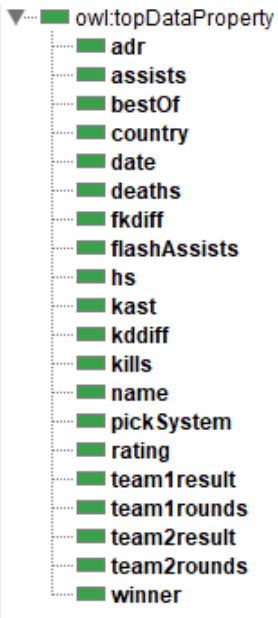

Fig. 3. Ontology data properties

In terms of restrictions, few were added. The number of "matchMaps" played in a match should never exceed 5. Regarding the number of match players, we considered adding an upper bound of 10 (5 for each team), but eventually realized that it would not make sense since it has become a recent trend for teams to use a roster of more than 5 players [40].

D. Ontology Population

Protegé's installation bundle includes the Cellfie plugin - a tool for importing spreadsheet data into OWL ontologies [41]. This proved to be an ideal solution for two reasons: there would be no need to find third-party plugins, and our dataset is composed of CSV files (meaning it would take very little additional effort to parse the data into an acceptable format). In fact, all we had to do was import the files into an Excel workbook, where each file becomes a worksheet. Then, through a process of trial-and-error, we remove the least relevant columns, and select a small number of rows for each sheet. This is due to the fact that Protegé does not deal well with large files, often throwing "out of memory" errors. Thus, we ended up with a workbook containing data for a set of matches that took place in January 2020. Next, to import this workbook into Protegé, we need to define a set of transformation rules: these rules identify how to create individuals and their properties from each row of the worksheets. These transformation rules can be found in the source code accompanying this article.

E. Ontology Exploration

In order to test our implementation, several SPARQL queries were constructed. It should be noted that, given the poor performance of Protegé when dealing with imports of large amounts of data, we will only consider some games from January 2020. This will hinder the potential of some of the queries, as their relevance is greater when the timeframe in question is considerably longer. For example, querying about the competitions played by a certain team would benefit from data spanning a number of years.

1) How many different competitions were played by Heroic?: CS:GO can have a difficult schedule for teams, with many tournament organizers competing for the public interest. In some situations, teams even find themselves playing in the group stage of a tournament in the day after playing the final for another. Thus, it would be interesting to know in how many tournaments Heroic played in the span of a month. The results for this query can be seen in Table XIII.

```
SELECT distinct ?eventname
WHERE
{
  ?team :name "Heroic".
  ?match :event ?event.
  ?event :name ?eventname
  {
    ?match :team1 ?team
  }
  UNION
  {
    ?match :team2 ?team
  }
}
```

2) Which team picks the most maps they lose on?: In Counter-Strike, it is important to have a wide map pool, meaning that the team can have a strong performance in many situations. It is also important to be confident in the map the team picks, since it theoretically should be one of their best. So, what teams should reconsider their pick? The results for this query can be seen in Table XIV.

```
SELECT ?team (count(?matchmap) as ?maps)
WHERE
{
  ?pick :match ?match.
  ?pick rdf:type :Picks.
  ?matchmap :match ?match.
  ?matchmap :map ?map.
  {
    ?match :team1 ?team.
    ?pick :team1Picked1 ?map.
    ?matchmap :team1rounds ?rounds.
  }
  UNION
  {
    ?match :team2 ?team.
    ?pick :team2Picked1 ?map.
    ?matchmap :team2rounds ?rounds.
  }
  FILTER(?rounds < 16)
}
GROUP BY ?team
ORDER BY DESC (?maps)
LIMIT 10
```

3) Which team had the most wins?: Winning is what every player or team strives for. One could assume that the teams that win most games are the best, however, some contextualization is necessary: if a team only plays against lower tiered teams, then it's only natural that they win frequently. It's an interesting query, nonetheless, and the results can be seen in Table XV.

```
SELECT ?team (count(?match) as ?wins)
WHERE
{
  ?match :date ?date.
  {
    ?match :team1 ?team.
    ?match :winner 1
  }
  UNION
  {
    ?match :team2 ?team.
    ?match :winner 2
  }
  FILTER (?date > "2020-01-01T00:00:00Z"^^xsd:
    dateTime && ?date <  "2021-01-01T00:00:00Z"^^xsd
    :dateTime)
}
GROUP BY ?team
ORDER BY DESC (?wins)
LIMIT 10
```

4) Against which team did Heroic lose the most?: Sometimes, even the strongest teams have touch opponents who they can't figure out how to beat. The results for this query can be seen in Table XVI.

```
SELECT ?opponent (count(?match) as ?losses)
WHERE
{
  ?team :name "Heroic".
  ?match :date ?date.
  {
    ?match :team1 ?team.
    ?match :team2 ?opponent.
    ?match :winner 2
  }
  UNION
  {
    ?match :team2 ?team.
    ?match :team1 ?opponent.
    ?match :winner 1
  }
  FILTER (?date > "2020-01-01T00:00:00Z"^^xsd:
    dateTime && ?date <  "2021-01-01T00:00:00Z"^^xsd
    :dateTime)
}
GROUP BY ?opponent
ORDER BY DESC (?losses)
LIMIT 10
```

5) What map does Heroic ban first the most?: As explained before, teams should have a wide map pool, however it is very difficult to be proficient in every map, thus teams decide to take advantage of the veto process and remove their worst map straight away. The results for this query can be seen in Table XVII.

```
SELECT ?firstban   (count(?match) as ?matches)
WHERE
{
  ?pick :match ?match.
  ?pick rdf:type :Picks.
  ?team :name "Heroic".
  {
    ?match :team1 ?team.
    ?pick :team1Removed1 ?firstban.
  }
  UNION
  {
      ?match :team2 ?team.
      ?pick :team2Removed1 ?firstban.
  }
}
GROUP BY ?firstban
ORDER BY DESC (?matches)
```

6) Who is cadiaN's most difficult opponent?: Sometimes, players performances decrease significantly when playing against certain opponents, and such performance is reflected in the player's rating: when it is lower than 1, it means that it is below average. The results for this query can be seen in Table XVIII.

```
SELECT ?opponent (count(?match) as ?matches)
WHERE
{
  ?player :name "cadiaN".
  ?matchplayer :playerRef ?player.
  ?matchplayer :team ?team.
  ?matchplayer :match ?match.
  ?matchplayer :rating ?rating.
  {
    ?match :team1 ?team.
    ?match :team2 ?opponent.
  }
  UNION
  {
    ?match :team2 ?team.
    ?match :team1 ?opponent.
  }
  FILTER (?rating < 1.0).
}
GROUP BY ?opponent
ORDER BY DESC (?matches)
LIMIT 10
```

7) How many times has Heroic won less than 10 rounds in a match map? : Usually a game is considered close if both teams achieved more than 10 rounds. The results for this query can be seen in Table XIX.

```
SELECT ?map ?opponent ?roundswon ?date
WHERE
{
  ?team :name "Heroic".
  ?matchmap :match ?match.
  ?matchmap :map ?map.
  ?match :date ?date
  {
    ?match :team1 ?team.
    ?match :team2 ?opponent.
    ?matchmap :team1rounds ?roundswon.
  }
  UNION
  {
    ?match :team2 ?team.
    ?match :team1 ?opponent.
    ?matchmap :team2rounds ?roundswon.
  }
  FILTER(?roundswon < 10)
}
```

F. Tool Evaluation

Protegé provides a simple interface for the definition of classes, object properties and data properties. The integrated environment with the possibility of populating the ontology and perform SPARQL queries without installing external plugins is also very appreciated. Additionally, while not used at first, the reasoner provided handy explanations for inconsistencies in the ontology. Later in the development of this project, it also provided us with inferences, specifically regarding inverse object properties.

The biggest issue we found with the tool is related to an increased CPU and memory usage, which limited the number of individuals we could create in the population process. This ties into the analysis of the query results. As previously mentioned, we could only work with a subset of the data from a very limited period of January 2020. In the world of Counter-Strike, it is very difficult to draw conclusions from such a small period, as usually we deal with ranges from 3 to 6 months, or a full year. For instance, the query "How many times has Heroic won less than 10 rounds in a match map?" was created because the team Astralis, one of the best in the world in recent years, is notorious for being a very difficult team to beat. Therefore, we wanted to study what teams were able to beat them convincingly. However, Astralis didn't play a single match in the considered timeframe, so we had to change teams.

G. Information Retrieval vs Semantic Web

The biggest differences between Information Retrieval and Semantic Web tools is the type of data being used, and the relevance of the results returned from queries. In IR, we mostly deal with unstructured, textual data and we can't guarantee that every document returned for a query is relevant towards the information need. In Semantic Web, we work with structured data and the results returned from SPARQL queries are always relevant (unless there is some mistake in the query).

In the context of Counter-Strike, querying over structured data is a better way of obtaining results when we are dealing with concrete terms. Let's look at the first information need in section IV-D "Matches played between Astralis and Natus Vincere during BLAST tournaments": obtaining the results to this information need would be much easier and accurate using semantic web tools, since we only need to relate the team names and tournament name. Looking at the third query in section V-E "Which team had the most wins", we can conclude that the results with IR tools would be worse, since we already saw previously that it is hard to determine from textual search which team won a match.

However, IR tools still provide value when we are dealing with more abstract information. For example, the information need "Matches were FURIA's aggressiveness was highlighted", which could also be conveyed as "Matches where FURIA were aggressive" would be difficult to retrieve with semantic web tools, since this isn't something we can clearly reason over with the available statistics.

H. Applications

The CS:GO ontology could be an opportunity for HLTV (the source of the data) to augment their data, so other platforms can understand and use it, thus increasing the shared knowledge in the field. One of the main applications would be using the ontology to provide news coverage for large events, similar to how BBC used the Sport Ontology introduced above to cover the 2010 Football World Cup.

One other possible approach would be to adapt this ontology to encompass, in a broader sense, games played in the Esports scene. While some of the mechanics coded in the data and object properties are quite specific to CS:GO, its overarching concepts can easily be related to other games (such as League of Legends and Valorant, two extremely popular titles in the competitive contexts).

VI. Conclusions

This paper addressed the collection and characterization of data related to CS:GO Professional Matches and implementation of a search system that can provide the user with a compreensive view of said data in an information retrieval context. Two datasets were analysed, one for the match statistics and another for the CS:GO-related news. The datasets were cleaned, refined, and subsequently characterized, both conceptually and statistically. A search system was then built upon the document collection and evaluated under different configurations. Finally, an ontology was built and populated with a subset of our data and used to perform some relevant queries.

References

[1] O. Scott. November 27, 2000. "Half-Life: Counter-Strike Review". *Gamespot*. [Online]. Available : https://www.gamespot.com/reviews/half-life-counter-strike-review/1900-2657769/. [Accessed: October 25, 2020]

[2] Valve Corporation. 2020. "Counter-Strike: Global Offensive". [Online]. Available : https://store.steampowered.com/app/730/CounterStrike_Global_Offensive/. [Accessed: October 25, 2020]

[3] SteamDB. 2020. "Counter-Strike: Global Offensive". [Online]. Available : https://steamdb.info/app/730/graphs/. [Accessed: October 25, 2020]

[4] Liquipedia. 4 September, 2020. “World Electronic Sports Games 2016”. [Online]. Available : https://liquipedia.net/counterstrike/World_Electronic_Sports_Games/2016. [Accessed: October 25, 2020]
[5] Field Level Media. March 2, 2020. “IEM Katowice sets viewership record amid coronavirus outbreak”. *Reuters*. [Online]. Available : https://www.reuters.com/article/esports-csgo-katowice-viewership-idUSFLM3NTQY6. [Accessed: October 25, 2020]
[6] J. Hamari, M. Sjöblom. 2017. “What is eSports and why do people watch it?”. *Internet Research*, vol. 27, no. 2. pp. 2.
[7] Kaggle. 2020. “Kaggle: Your Machine Learning And Data Science Community”. [Online]. Available: https://www.kaggle.com/. [Accessed: October 25, 2020].
[8] Siteworthtraffic. 2020. [Online]. Available: https://www.siteworthtraffic.com/report/hltv.org. [Accessed: October 26, 2020].
[9] ELEAGUE. September 27, 2016. “ELEAGUE to Host CS:GO Major Championship”. [Online]. Available: https://www.eleague.com/news/2016/9/27/eleague-to-host-csgo-major-championship. [Accessed: October 26, 2020].
[10] LucasAM. August 2, 2019. “CSPPA announce 2020 summer player break dates”. [Online]. Available: https://www.hltv.org/news/27517/csppa-announce-2020-summer-player-break-dates. [Accessed: October 26, 2020].
[11] MIRAA. October 11, 2020. “Flashpoint 2 closed qualifier invites revealed”. [Online]. Available: https://www.hltv.org/news/30443/flashpoint-2-closed-qualifier-invites-revealed. [Accessed: October 26, 2020].
[12] Counter-Strike: Global Offensive. [Online]. Available: https://blog.counter-strike.net/index.php/2019/05/24172/. [Accessed: October 26, 2020].
[13] HLTV. 2020. “CS:GO News & Coverage”. [Online]. Available: https://hltv.org. [Accessed: October 26, 2020].
[14] OpenRefine. 2020. “OpenRefine”. [Online]. Available: https://openrefine.org/. [Accessed: October 26, 2020].
[15] Scrapy. 2020. “A Fast and Powerful Scraping and Web Crawling Framework”. [Online]. Available: https://scrapy.org/. [Accessed: October 26, 2020].
[16] spaCy. 2020. “Industrial-Strength Natural Language Processing”. [Online]. Available: https://spacy.io/. [Accessed: October 26, 2020].
[17] HLTV. 2020. “Natus Vincere vs. NiP at BLAST Premier Fall Series 2020 — HLTV.org”. [Online]. Available: https://www.hltv.org/matches/2344817/natus-vincere-vs-nip-blast-premier-fall-series-2020. [Accessed: October 26, 2020].
[18] HLTV. 2020. “BLAST Premier Fall Series 2020”. [Online]. Available: https://www.hltv.org/stats/matches/heatmap/mapstatsid/110809/nip-vs-natus-vincere?showKills=true&showDeaths=false&firstKillsOnly=false&allowEmpty=false&showKillDataset=true&showDeathDataset=false. [Accessed: October 26, 2020].
[19] Hawka. August 2, 2020. “aRT: The Most Aggressive Player In CS:GO History”. [Online]. Available: https://www.youtube.com/watch?v=s9MbqpTnOh4. [Accessed: October 26, 2020].
[20] Leetify. 2020. “Leetify - CS:GO Stats & Actionable Insights to help you improve”. [Online]. Available: https://leetify.com/. [Accessed: October 26, 2020].
[21] Professeur. September 1, 2020. “Heroic suspend hunden following ban”. [Online]. Available: https://www.hltv.org/news/30225/heroic-suspend-hunden-following-ban. [Accessed: October 26, 2020].
[22] LucasAM. October 22, 2020. “C0NTACT ADD RIGON AND SPINX”. [Online]. Available: https://www.hltv.org/news/30509/c0ntact-add-rigon-and-spinx. [Accessed: October 26, 2020].
[23] Liquipedia. 2020. “Liquipedia Counter-Strike Wiki”. [Online]. Available: https://liquipedia.net/counterstrike/Main_Page. [Accessed: October 26, 2020].
[24] M. Machado. 2020. “CS:GO Professional Matches”(Version 1). [Online]. Available: https://www.kaggle.com/mateusdmachado/csgo-professional-matches. [Accessed: October 26, 2020].
[25] HLTV. 2020. “HLTV.org Terms”. [Online]. Available: https://www.hltv.org/terms. [Accessed: October 26, 2020].
[26] Christopher D. Manning, Hinrich Schütze, and Prabhakar Raghavan. 2008. Introduction to Information Retrieval. Cambridge University Press.
[27] Emma Matthews. September 28, 2020. “37 CS:GO coaches have been banned for abusing the Spectator bug”. [Online]. Available: https://www.pcgamer.com/37-csgo-coaches-have-been-banned-for-abusing-the-spectator-bug/. [Accessed: November 25, 2020]
[28] The Apache Software Foundation. 2020. “Apache Solr”. [Online]. Available: https://lucene.apache.org/solr/. [Accessed: November 25, 2020]
[29] Elasticsearch B.V. 2020. “Elasticsearch: The Official Distributed Search & Analytics Engine”. [Online]. Available: https://www.elastic.co/Elasticsearch/. [Accessed: November 25, 2020]
[30] DB-Engines. 2020. “DB-Engines Ranking of Search Engines”. [Online]. Available: https://db-engines.com/en/ranking/search+engine. [Accessed: November 25, 2020]
[31] The Apache Software Foundation. 2020. “Welcome to Apache Lucene”. [Online]. Available at: https://lucene.apache.org/. [Accessed: November 25, 2020]
[32] Asaf Yigal. 2020. “Solr vs. Elasticsearch: Who’s The Leading Open Source Search Engine?”. [Online]. Available at: https://logz.io/blog/solr-vs-Elasticsearch/. [Accessed: November 25, 2020]
[33] Elasticsearch B.V. 2020. “Beats: Data Shippers for Elasticsearch”. [Online]. Available at: https://www.elastic.co/beats/. [Accessed: November 25, 2020]
[34] Stack Overflow. 2020. “Solr Nested Documents not properly setup”. [Online]. Available at: https://stackoverflow.com/questions/59566421/solr-nested-documents-not-properly-setup. [Accessed: November 25, 2020]
[35] Steve Rowe. April 18, 2017. “ Multi-Word Synonyms in Solr With Query-Time Support”. [Online]. Available at: https://lucidworks.com/post/multi-word-synonyms-solr-adds-query-time-support. [Accessed: November 25, 2020]
[36] Martin Porter. 1980. “An algorithm for suffix stripping”. Program, vol. 14 no. 3, pp. 130-7.
[37] Berners-Lee, Tim, James Hendler, and Ora Lassila. ”The semantic web.” Scientific american 284, no. 5 (2001): 34-4
[38] “Ontologies - Sport Ontology.” BBC. BBC. [Accessed: January 7, 2021]. https://www.bbc.co.uk/ontologies/sport.
[39] “BBC Sport.” Epimorphics, September 29, 2020. https://www.epimorphics.com/casestudy/bbc-sport/.
[40] Professeur. October 16, 2020. ‘OFFICIAL: NIVERA JOINS VITALITY”. [Online]. Available:https://www.hltv.org/news/30477/official-nivera-joins-vitality. [Accessed: January 7, 2021].
[41] Protegeproject. “Protegeproject/Cellfie-Plugin.” GitHub. [Accessed: January 7, 2021]. https://github.com/protegeproject/cellfie-plugin.
[42] Janne Parkkila, Filip Radulovic, María Poveda, and Daniel Garijo. December 2014. “The Video Game Ontology”. [Online]. Available at: http://vocab.linkeddata.es/vgo/. [Accessed: January 7, 2021].

APPENDIX A
GLOSSARY

- **Eco round** - in such round, the team chooses to save money by not buying weapons / grenades. Usually this is done with the purpose of having better weapons in the next round;
- **Force buy** - round where the team chooses to spend all of their money despite not having enough money for the theoretically better weapons;
- **Full buy** - the team has enough money to buy the best weapons and grenades;
- **Veto** - CS:GO has 7 active maps. Most professional games have a best of 1, 3, or 5 format, and therefore there needs to be picks and bans, where teams choose which maps they wish, or not, to play;
- **T/CT** - CS:GO has two teams playing against each other on opposite sides. CTs are meant to defend the A and B bomb sites, while Ts want to plant and explode a C4 on one of said bomb sites. After 15 rounds, they will swap. First to 16 rounds wins;

APPENDIX B
TABLES

TABLE X
DATASET FILES

Collection	Format	Function
Economy	CSV	Money earned by each team in all rounds of a given map
News	CSV	News collected from HLTV.org
News Entities	CSV	Entities extracted from the scraped news
Picks	CSV	Maps chosen and excluded by the teams in a given match
Players	CSV	Statistics for a given player in a given match
Results	CSV	Statistics for both teams in a given round

TABLE XI
SEARCH TASKS

Search for	Restrict on	Order by
Player	Kills Assists Deaths HS Flash Assists KAST KD ADR FKDIFF Rating	Map Team Against Date Interval Side (T/CT) Team Nationality
Teams	Wins Games Played Round Win % Force Buy % Upset Potential Pick Win Rate	Map Team Against Date Interval Side (T/CT)
Matches	Date	Map Teams Date Interval Event
News	Date	Date Interval

APPENDIX C
GRAPHICS

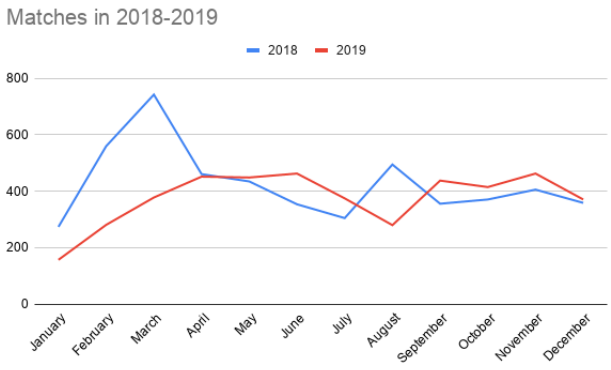

Fig. 4. Matches played in 2018-2019

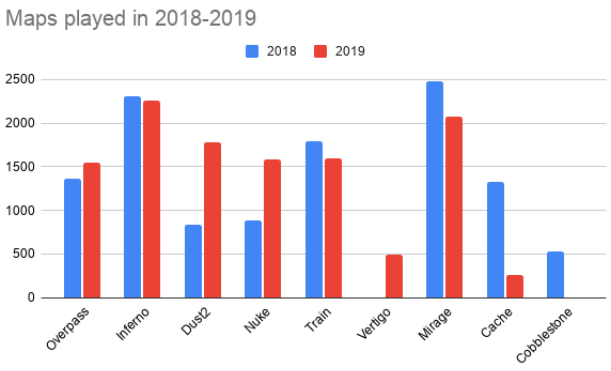

Fig. 5. Maps played in 2018 and 2019

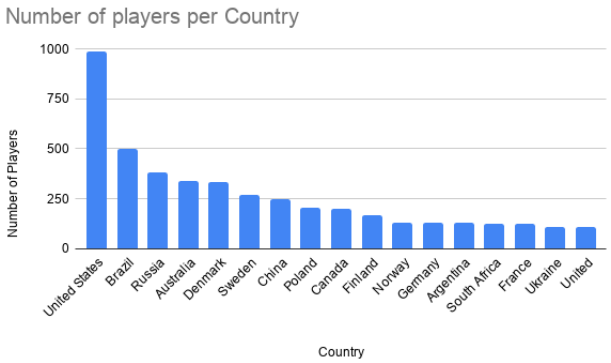

Fig. 6. Number of players per country in 2018-2019

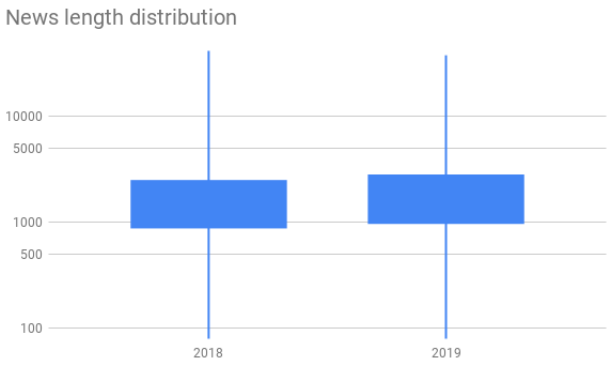

Fig. 7. Number of characters in news in 2018-2019

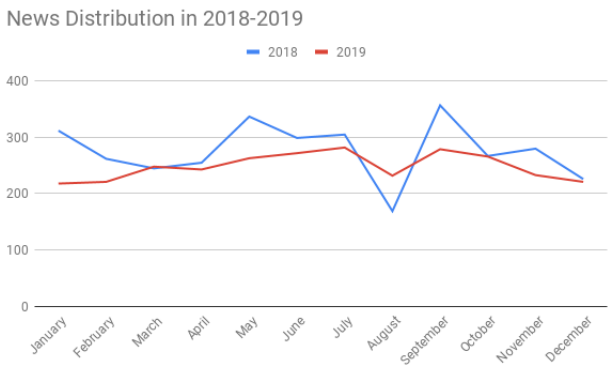

Fig. 8. News distribution in 2018-2019

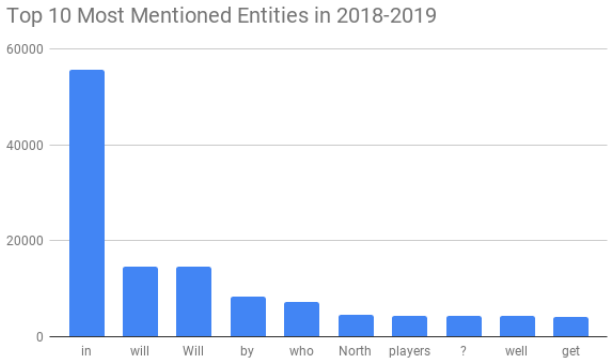

Fig. 9. Top 10 Entities in 2018 and 2019

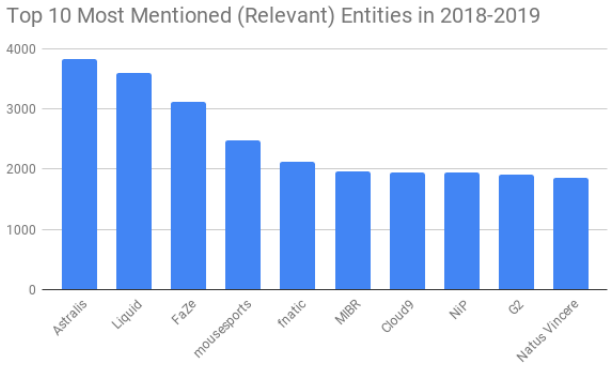

Fig. 10. Top 10 (Relevant) Entities in 2018 and 2019

APPENDIX D
DATASET FILE STRUCTURE

TABLE XII
FILE STRUCTURE

File Name	Columns
Economy	date match_id event_id team_1 team_2 best_of map t1_start t2_start
Players	date player_name team opponent country player_id match_id event_id event_name best_of
Picks	date team_1 team_2 inverted_teams match_id event_id best_of system t1_removed_1 t1_removed_2
Results	date team_2 team_2 map result_1 result_2 map_winner starting_ct ct_1 t_2

APPENDIX E
DESCRIPTION OF COUNTER-STRIKE: GLOBAL OFFENSIVE MECHANICS

CS:GO is the most popular shooting game in the market, mainly because of its simple yet hooking mechanics. The player starts the game as part of a team of 5, and is either a Terrorist or Counter-Terrorist. In each round, the terrorists must try to plant and explode a bomb in one of two bombsites or kill the other team, while the counter-terrorists have to stop their adversaries, by killing them or defusing the bomb. This goes on for 15 rounds in the first half, then the teams switch sides: first to 16 rounds win.

In order to kill opponents, players need to buy weapons at the beginning of the round. They must choose carefully what to buy since their money is limited, and as expected the better weapons are more expensive. Apart from this, players can also purchase equipment such as armour, defuse kits and grenades (smoke, flashbang, high explosive or incendiary), in order to gain situational advantages.

The economy is one of the main problems in CS:GO. If a team doesn't know how to manage it, they'll most likely lose the match. In the first round of the game, each player starts with a pistol and 800 dollars. This is enough money to buy light armour, grenades or an upgraded pistol. When a team wins a round, they'll receive around 3000 dollars. When they lose, they still receive a bonus, that increases with each

consecutive round that is lost. However, when a losing streak is broken, the bonus for losing future rounds decreases as well. Therefore, depending on the situation, teams may have to make tough decisions. If they're winning a lot of rounds their economy should be great (unless they can't survive the rounds with more than 2 players alive, in that case they have to keep rebuying equipment). When they lose a round, they have to see how much money they have, and decide whether they should buy or not. Additionally, they may also need to make mid-round decisions: if a player is, for example, in a 1 versus 4 situation, he'll most likely decide to save his equipment into the next round, while the opposing team may try to hunt him so that he can't keep anything.

Finally, each game of Counter-Strike takes place in a map: a contained world, that normally contains two locations on opposite sides where the teams spawn in each round, and two bombsites where the terrorists try to plant a bomb. Once again, the concept is simple, yet most of the maps have their own identity, something that separates them from others, be it their layout, geographical/historical context, sound queues or even colors.

APPENDIX F
SPARQL QUERIES RESULTS

TABLE XIII
Q1 RESULTS

Name
DreamHack Open Anaheim 2020 Europe Closed Qualifier
DreamHack Open Leipzig 2020
DreamHack Open Leipzig 2020 Europe Closed Qualifier
IEM Katowice 2020 Europe Closed Qualifier

TABLE XIV
Q2 RESULTS

Team	Maps
SKADE	7
Heroic	6
ARCY	4
RiotSquad	4
Chaos	3
Brute	3
MADLions	3
NewEnglandWhalers	3
August	3

TABLE XV
Q3 RESULTS

Team	Wins
MADLions	10
ex-Genuine	9
Heroic	8
SKADE	7
CopenhagenFlames	7
HAVU	6
INTZ	6
GambitYoungsters	6
GamerLegion	6
BIG	6

TABLE XVI
Q4 RESULTS

Opponent	Losses
MADLions	2
forZe	1
AGO	1
BIG	1

TABLE XVII
Q5 RESULTS

Firstban	Matches
Vertigo	3
Mirage	2
Train	2
Inferno	2
Overpass	1
Nuke	1
Dust2	1

TABLE XVIII
Q6 RESULTS

Opponent	Matches
BIG	1
forZe	1
AGO	1

TABLE XIX
Q7 RESULTS

Map	Opponent	Roundswon	Date
Nuke	forZe	4	19/01/2020
Inferno	forZe	9	19/01/2020
Vertigo	GODSENT	5	19/01/2020
Train	AGO	3	9/01/2020
Vertigo	AGO	5	9/01/2020
Train	MADLions	7	9/01/2020
Overpass	BIG	9	26/01/2020

APPENDIX G
DATA PIPELINE

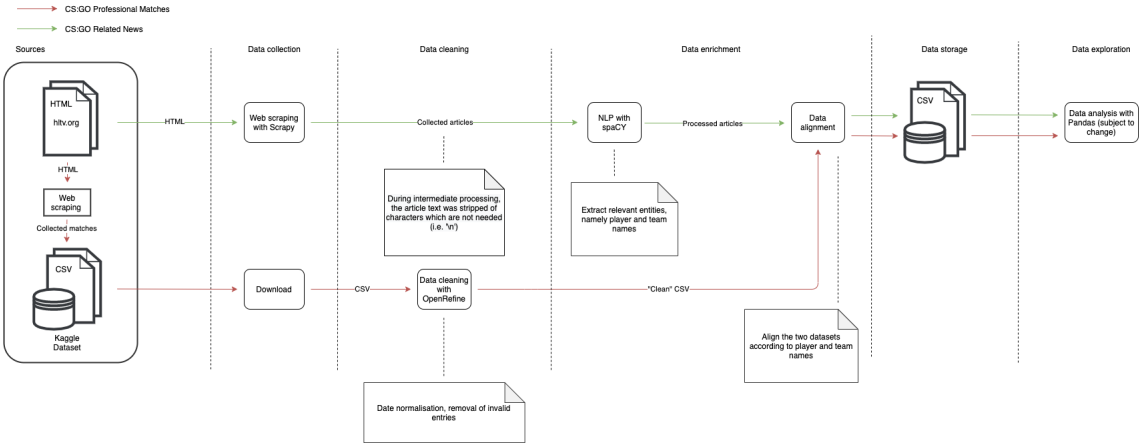

Fig. 11. Data pipeline

APPENDIX H
CONCEPTUAL MODEL

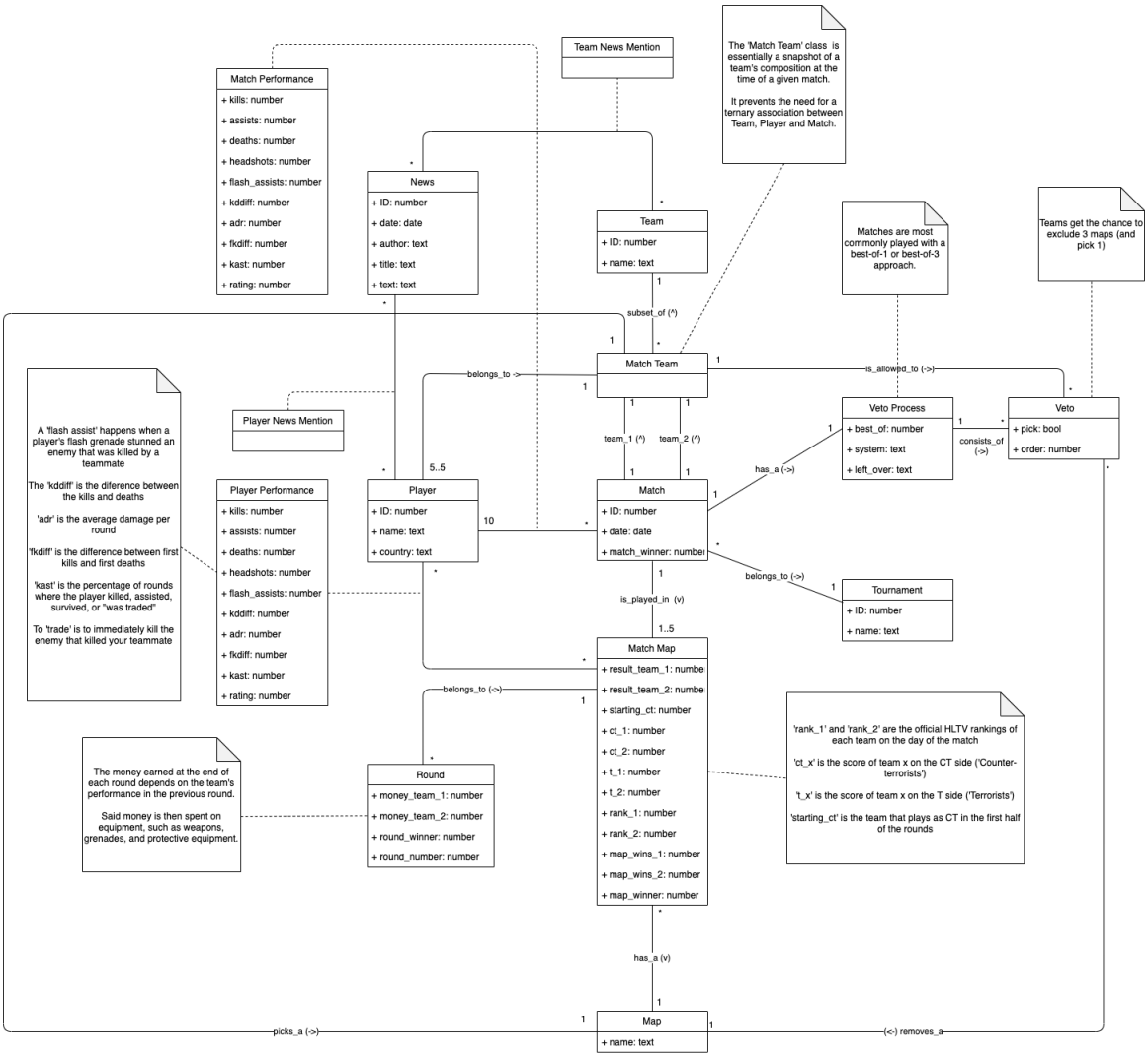

Fig. 12. Conceptual model