A Practical Proposal for Building Counter-Strike Global Offensive Datasets

Erick Rocha
Instituto de Computação
UFRJ
Rio de Janeiro, Brasil
erickkrocha@gmail.com

Henrique Maio
Escola de Engenharia
UFRJ
Rio de Janeiro, Brasil
henriquemaio@poli.ufrj.br

Daniel S. Menasché
Instituto de Computação
UFRJ
Rio de Janeiro, Brasil
sadoc@ic.ufrj.br

Claudio Miceli
Escola de Engenharia
UFRJ
Rio de Janeiro, Brasil
miceli@nce.ufrj.br

***Resumo*—There is a growing necessity for insightful and meaningful analytics within eSports: be it to entertain spectators as they watch their favorite teams compete, to automatically identify and catch cheaters or even to gain a competitive edge over an opponent, there is a plethora of potential applications for analytics within the scene. It follows then, that there is also a necessity for well structured and organized datasets that enable efficient data exploration and serve as the foundation for the visualization and analytics layers. Because of this, the entire process - from data collection at the source to the means of accessing the desired information - need to be planned out to address those needs. Our work provides the means by which to construct such a dataset for the Counter-Strike Global Offensive (CS:GO) game, thus opening up a range of possible applications on top of the data.**

***Palavras-chave*—counter-strike, esport, data, dataset, insert**

I. Introduction

With the development and evolution of technology, the same atmosphere that has ignited crowds of spectators and inspired people all over the world in sports [1], has also become part of electronic games. By breaking the frontiers and allowing people to not only spectate, but also compete with other people around the world [2], competitions that until around the 2000s were mostly among amateur players, started to attract the market's attention. Rapidly following it, a large structure was formed around professional competitions which became collectively known as "eSports" [3].

According to SuperData's anual report [4], digital games earned $126.6B in 2020 and many organizations have been investing on the creation of teams and leagues to explore that potential. One of the games that have for long now been a pillar of the eSports world and that greatly influenced in its growth is the franchise of the multiplayer first-person shooter: Counter-Strike (CS).

Released in 1999 as a modification[1] for another game called Half-Life, the Counter-Strike franchise has surpassed and outlived its predecessor with many versions of the game being developed and released: over the last 12 years, it has been one of the most played games in the world, having sold more than 25 million copies. Its latest version is called "Counter-Strike: Global Offensive" (CS:GO) and is the subject of this article.

[1]Modifications are also known as mods in the gaming jargon.

On the tail of the incredible growth of the eSports market and of the CS:GO competitive scenario is the ever growing need for insightful analytics and visualizations. Acquiring data to support these goals is not easy: the readily available data one can find in most CS:GO websites can only support basic analysis. Furthermore, there are many players[2][3] in the market that seemingly employ sophisticated data retrieval strategies to obtain quality data, but the knowledge and tooling used to obtain such data seems to be intellectual property. This ends up leaving a gap with respect to publicly available datasets.

Thus, with hopes of enabling data exploration and ultimately empowering the community to build meaningful and insightful analysis, this article aims to propose a strategy and architecture that can be leveraged to retrieve quality CS:GO data.

Outline. The rest of this article is structured as follows. Section II presents the landscape, followed by challenges in Section III. Sections IV and V describe the strategy to build the dataset and the subsumed architecture. The dataset is described in Section VI, related work in Section VII, dataset availability in Section VIII and Section IX concludes.

II. Landscape

There are two types of sources one can leverage to obtain CS:GO data.

The first source – henceforth referred to as a source of "high level data" – is any website or application that tracks competitive matches and displays their results. It offers information such as the date window in which championships took place, the dates in which matches were played, the teams and players that participated on those matches, their scores and finally, some high level statistics that can support only shallow analysis.

The second source – henceforth referred to as a source of "low level data" – is an actual "replay" file of a match. These "replay" files (called "demo" files by the community) are protobuf-serialized files that contain every important event that occurred within the match. It offers a plethora of information regarding every player at almost any given moment in the game, such as their positions, their net worth (i.e. current

[2]https://sixteenzero.net/
[3]https://csgostats.gg/

money plus investment in current equipment) and their actions (e.g. movements, shots, grenades thrown).

III. Challenges

Collecting data in a large scale fashion can be a problem even if the data is readily available: there are many avenues of concern within the overall process of collecting, storing and shaping raw data into valuable information. For instance, how do you optimize and scale your data fetching routine when the source isn't well structured or protects itself against crawlers? How do you store the data while minimizing infrastructure costs and maximizing retrieval response time? Will the data require real time aggregation? Should it be structured with a relational schema or not? There are many more questions looming over the topic and answering these questions requires a deeper understanding of one's objectives when handling the data, which means it's not trivial.

The collection process for CS:GO data is no different. There are many challenges in capturing and structuring the data in a useful way. To name a few:

- Surpassing anti-crawling mechanisms
- Scaling the data fetching routine
- Extracting information out of CS:GO "demo" files
- Linking low and high level data (see Section II)
- Managing associated costs

The breadth and depth of data available about CS:GO is astounding. Once the challenges are overturned, it is possible to build a robust dataset that can be leveraged in many ways, such as to gain insight into characteristics of the game that are hidden from the "naked eye", to define new models that better evaluate the impact of each player in the game and to find trends in the game.[4]

IV. Strategy

To build a truly meaningful CS:GO dataset, we believe one has to leverage aspects from both high and low-level data sources. Recall from Section II that we distinguish two sorts of data:

- high-level data sources: championships, matches, maps, teams and players;
- low-level data sources: in-game actions such as player coordinates, shots fired and grenades thrown.

Thus, for the overall success of our endeavor, it was paramount to select a source that made it possible to capture both high and low-level data. Consider the HLTV CS:GO portal[5]: in it, one can not only browse championships and select the results of the matches played in them, but also download the "replay" files of each match.

Our overall strategy then consists of the following steps:

1) Retrieve high-level data of each match along with the low-level "demo" file
2) Retrieve low-level data from each downloaded "demo" file

[4]Such trends also known as "meta" within the gaming community.
[5]http://hltv.org

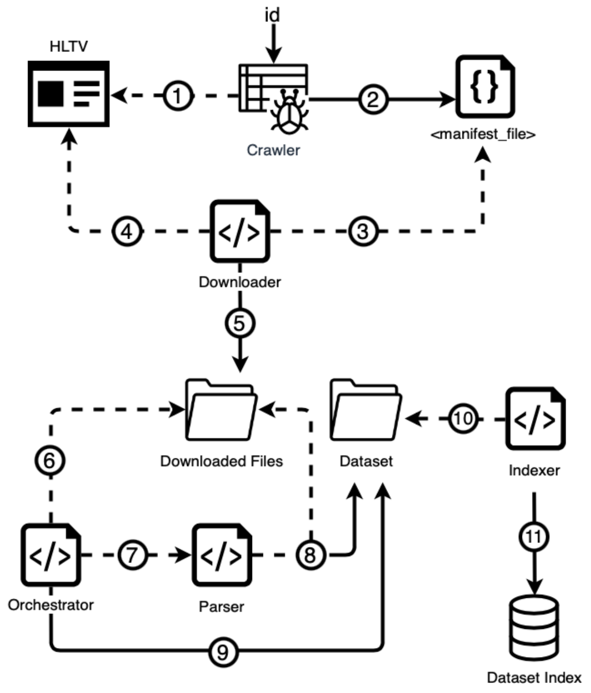

Fig. 1. Dataset generation process

3) Store the retrieved information
4) Expose stored information through a "queryable" interface

V. Architecture

Next, we describe our architecture to collect and structure high and low-level information about CS:GO championships. The designed architecture attempts to employ the following strategy while solving or working around the previously mentioned challenges. It consists of five different components:

1) A web crawler (for the HLTV[6] site)
2) A demo file downloader
3) A demo file parser
4) A parsing orchestrator
5) An indexer

A. Crawler

The web crawler is the entry point of the architecture: given the identification number of a championship, it crawls the HLTV website (1) and generates a manifest file (2) containing the high-level data of each match played in the championship along with the URLs to download the demo files.

B. Downloader

The demo file downloader consumes the manifest file (3) to retrieve the URLs and then proceeds to download the files (4), saving them to the configured path on the file system (5).

[6]See footnote 5.

C. Orchestrator

The orchestrator is the component responsible for consuming the downloaded files and generating the dataset. It starts by crawling its configured input directory and decompressing the downloaded files (6). For each decompressed demo file, it invokes the parser (7) and converts the parsed CSV files into the parquet format (9).

D. Parser

The demo file parser parses the downloaded protobuf-serialized demo files and writes the desired data in a set of CSV files (8).

E. Indexer

The indexer is the final piece of the puzzle: it indexes the dataset in the local file system (10) and generates an SQLite database file (11) that serves as an index for the low-level data.

VI. Dataset

Next, we describe the obtained dataset. We restrict our description, focusing on a proof-of-concept of what can be achieved by leveraging the architecture described in the previous section.

Our dataset contains all matches played in the "ESL Pro League Season 13" [7] championship, where 73 distinct matches were disputed throughout the competition, resulting in a total of 173 maps played.

A. Structure

Our dataset was designed to be consumed locally while minimizing the resources required to store and consume it. The main drivers behind the design were to allow cost-effective data exploration and to facilitate integration with well established means of local file consumption for data-science ends (e.g. through Pandas).

The dataset is structured as a set of files on the local file system and is composed of a manifest, an SQLite database file and a set of parquet files.

1) Manifest File: The manifest is a file containing high-level data, such as which teams and players competed in each match, which maps were played in it and when the match took place. This holds special importance as low-level data – such as the nickname or id of a player within a match – can change from match to match. In addition, it can be hard to map the in-game avatars to the real world people playing those avatars.

By linking this information, one can unlock the potential of the dataset and track metrics of a player across time, even if the player switched teams.

2) SQLite File: The SQLite database file is generated by reading the manifest files containing high-level information about every championship found when traversing the dataset root folder. It is an index of the data available within the dataset and its ultimate goal is to provide a queryable interface through which one can find the relevant parquet files with low-level information one wants to explore. In other words, this database bridges the gap between the high-level and low-level entities.

3) Parquet Files: The parquet files contain the low-level data, such as the events that occurred within the game and the positions of each player at virtually every moment of the game.

B. Required Resources

As previously mentioned, our architecture leverages the Apache Parquet format[8] as a means to decrease system resources when storing and consuming the dataset. The columnar nature of this format allows us to:

1) Apply column-specific compression and encoding schemes, drastically reducing the required disk size to hold the dataset;
2) Read only the desired columns when consuming the dataset, again drastically reducing the amount of memory required to hold the dataset in memory.

A quick comparison between the dataset in CSV format and the currently employed optimized parquet format shows the drastic difference in disk size required to store the dataset in the hard drive (see Table I).

TABLE I
CSV vs Parquet

Unit	CSV	Apache Parquet
(GB)	83.2	9.86

C. Using the Dataset

The standard way of using the dataset is to first query the index database to locate the parquet files of interest and then proceed to load the columns of the desired subsets of data (i.e. tick, player_death, etc). This workflow is very malleable in the sense that both the SQLite database and the parquet files can be read using a wide range of libraries across several programming languages.

D. Usage Possibilities and Impact

Next, we discuss some of the various potential uses for the data. We refer to scenarios of interest and to research questions that can be elaborated or resolved from the availability of our dataset.

- **What is the behavior of users mobility?**
  From the user's movement data, it would be possible to analyze how the best players in the teams move and how this affects their results. This information can be used by

[7]https://www.hltv.org/events/5553/esl-pro-league-season-13

[8]https://parquet.apache.org/documentation/latest/

these same players to find their flaws and fix it or by opponents to overcome those players. *We envision that the vast literature on human mobility may be contrasted against players mobility, and that generative mobility models can be used for practicing purposes.*

- **What are the most adopted game strategies?** Based on the dataset data, it would be possible to categorize the different strategies used by the teams and how the teams coordinate their actions during the match. *We envision that both supervised and unsupervised machine learning tools may be used for clustering and classification purposes.*
- **What are the best strategies to win a match?** Based on the strategies, it would be possible to infer and compare which were used in the championship and find out which were the most victorious or most efficient against the best teams. *We envision that using reinforcement learning one may also use our dataset for training purposes, to shed insight into novel strategies.*

VII. Related work

Works related to game data analysis have been published in recent years [5], [6], which shows the importance of obtaining data in an efficient and structured way. There are different ways to perform data collection to perform this analysis. An article published on SBGames in 2020 [7] proposed a method to collect game data from videos, texture of videos on screen and players' actions.

Another relevant work was published in SBGames 2017 [8]. It presented a game data analysis focused on the World of Warcraft game. Although this article presents a different methodology for data collection, an analysis of the players' behavior could also be made from the data set generated by the proposal presented here.

Among works focusing on Counter Strike, the two more closely related to ours are [9] and [10]. In [9] the authors provide an open source platform to collect data from CS:GO. Their platform is complementary but different from ours. In particular, it does not bridge high-level and low-level data (see Section IV).

In [10] the authors analyze data from HLTV. Their goal is to assess and predict player performance. We envision that our work can be instrumental to reproduce and expand previous efforts such as those reported in [10].

VIII. Dataset and crawler availability

Our dataset is available at https://tinyurl.com/ncnsj8s6.

The modules comprising our crawling infrastructure are available as follows:

- Crawler: https://github.com/ErickRDev/csgo-demo-crawler
- Downloader: https://github.com/ErickRDev/csgo-demo-downloader
- Orchestrator: https://github.com/ErickRDev/csgo-demo-parser-orchestrator
- Parser: https://github.com/ErickRDev/csgo-demo-parser
- Indexer: https://github.com/ErickRDev/csgo-dataset-indexer

IX. Conclusion and future work

The increasing popularity of eSports has raised an interest in analyzing and visualizing relevant data within the domain. When it comes to CS:GO, there are websites with readily available high-level data that, when coupled with low-level data extracted from "replay" files, provide the necessary input to build a robust dataset that can be ultimately leveraged to satisfy the needs of the community.

In this paper we presented such a proof-of-concept dataset along with the means by which to collect and expand on the data. Our method crawls the relevant high-level data from the HLTV website, parses the protobuf-encoded "replay" files to extract low-level data and employs a simple, yet effective strategy to bridge the gap between them.

The presented architecture represents a huge opportunity to explore CS:GO data and can be used to investigate several scenarios and aspects of the game, to create analytic reports and visualizations about the matches, players and championships.

As future work related to the generation of the dataset, we envision a number of opportunities to improve the proposed pipeline and to foster a plug-and-play analysis of the collected data. In particular, we envision additional automation of the whole pipeline, from data collection to data storage without manual intervention.

Acknowledgement

This work was partially supported by CAPES, CNPq and FAPERJ under Grant E-26/203.215/2017 and Grant E-26/211.144/2019.

References

[1] J. El-Harami, "Entertainment and recreation in the classical world-tourism products," *J. Mgmt. & Sustainability*, vol. 5, p. 168, 2015.

[2] D. Williams, "Structure and competition in the us home video game industry," *International Journal on Media Management*, vol. 4, no. 1, pp. 41–54, 2002.

[3] J. G. Reitman, M. J. Anderson-Coto, M. Wu, J. S. Lee, and C. Steinkuehler, "Esports research: A literature review," *Games and Culture*, vol. 15, no. 1, pp. 32–50, 2020.

[4] SuperData, "Year in review: Digital games and interactive media," *https://g-mnews.com/en/digital-games-and-interactive-media-earnings-rose-12-to-us139-9-billion-in-2020/*, 2021.

[5] M. Varvello and G. M. Voelker, "Second life: a social network of humans and bots," in *Proceedings of the 20th international workshop on network and operating systems support for digital audio and video*, 2010, pp. 9–14.

[6] M. Varvello, F. Picconi, C. Diot, and E. Biersack, "Is there life in second life?" in *Proceedings of the 2008 ACM CoNEXT Conference*, 2008, pp. 1–12.

[7] E. S. Siqueira, C. D. Castanho, G. N. Rodrigues, and R. P. Jacobi, "A data analysis of player in world of warcraft using game data mining," in *2017 16th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. IEEE, 2017, pp. 1–9.

[8] N. T. Yada, J. Souza, and A. R. Ortoncelli, "Digital game video summarization based on screen and player videos," in *2020 16th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, 2020.

[9] P. Xenopoulos, H. Doraiswamy, and C. Silva, "Valuing player actions in counter-strike: Global offensive," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 1283–1292.

[10] D. Bednárek, M. Krulis, J. Yaghob, and F. Zavoral, "Data preprocessing of esport game records-counter-strike: Global offensive," *DOI: 10.5220/0006475002690276*, 2017.