Popularity of videogames on Twitch and Twitter

Khaled Hechmi[1], Gian Carlo Milanese[2]

Abstract
The gaming sector is no longer confined to local and online matches: social networks such as Twitch and Twitter are benefiting from the ever-growing success of the industry as they are the de facto virtual square where people from all around the world share their progress and discuss outstanding gameplay moves or the latest rumors. In this project we will try to measure and quantify the "buzz" around the most popular games of the past and present on these two social media, so that it will be possible to analyze in a data-driven way the popularity of the selected games.

Keywords
Gaming — Twitch — Twitter — MongoDB

[1] *793085, Università degli Studi di Milano Bicocca, CdLM Data Science*
[2] *848629, Università degli Studi di Milano Bicocca, CdLM Data Science*

Contents

Introduction

The goal of this project is to examine the popularity of a list of selected video games on Twitch and Twitter. After identifying a list of 50 games to keep track of, the official Twitch and Twitter APIs were used in order to collect the required data. This report will first go through the steps taken to collect, transform and store the data, and will then give an overview of some explorative queries and a visualization.

This report is organized as follows:

1. Data Collection Identification of the list of 50 games to monitor; description of the collected data, its sources, and the manner of collection.

2. Data Management Organization of the data in a MongoDB database.

3. Data Exploration Exploration of statistics of the data by querying the MongoDB database.

4. Data Visualization A visualization of the collected data.

1. Data Collection

1.1 Identifying the list of games

The first task has been to identify a list of 50 suitable games to collect data about. In order to do so, games that satisfy at least one of the following criteria were considered:

- the game has achieved very high global sales numbers;
- the game is, on average, one of the most streamed games on Twitch.

The idea is that a game that satisfies either of these two criteria should be popular enough to be streamed on Twitch and tweeted about, so that data about it can be collected. There is also an interest to see if the best selling games of all times are still relevant today or they have become a niche, played only by a handful of players.

Firstly, games with high sales numbers were identified by scraping the *VGChartz*, a website and industry research firm that publishes over 7,000 unique estimates per week relating to worldwide game hardware and software sales and hosts a game database with over 40,000 titles listed and 1.5 million unique data points [1]. The script used for downloading the VGChartz data is heavily based on the one made and shared on GitHub by GregorUT [2], even though the web-scraping logic was improved by adding plausible user agents and reducing the number of HTTP GET requests necessary for downloading all the ranked games: a Pull Request with all the improvements was created and is currently waiting for approval [3]. Table 1 shows a few lines of the downloaded data (for a subset of columns), which has been saved as a CSV file.

Rank	Name	Platform	Year	Publisher	Developer	Global_Sales
1	Wii Sports	Wii	2006	Nintendo	Nintendo EAD	82.65
2	Super Mario Bros.	NES	1985	Nintendo	Nintendo EAD	40.24
3	Mario Kart Wii	Wii	2008	Nintendo	Nintendo EAD	35.98
4	PLAYERUNKNOWN'S BATTLEGROUNDS	PC	2017	PUGB Corporation	PUGB Corporation	NA
5	Wii Sports Resort	Wii	2009	Nintendo	Nintendo EAD	32.9
6	Pokémon Red/Green/ Blue Version	GB	1998	Nintendo	Game Freak	31.37

Table 1. A few rows and columns of the VGChartz Data

After collecting the data, the total sales number for each title was computed by summing the global sales on each platform, and a list of top 25 best selling videogames was determined, excluding a few games that are not streamed on Twitch, namely *Brain Age, Kinect Adventures, Nintendogs, Wii Fit and Wii Fit Plus*. The following are some of the games featured in the resulting list:

- many *Call of Duty* games, such as *Black Ops* 1, 2 and 3, *Modern Warfare* 2 and 3, *Advanced Warfare* and *Ghosts*;
- *Grand Theft Auto San Andreas*, *IV* and *V*;
- *Mario Bros* games such as *Super Mario Bros, Super Mario World, New Super Mario Bros, New Super Mario Bros Wii*;
- *Mario Kart DS* and *Mario Kart Wii*;
- *Pokémon Red/Blue* and *Pokémon Gold/Silver*;
- *Tetris* (NES and GameBoy versions);
- Wii games like *Wii Sports*, *Wii Sports Resort* and *Wii Play*.

Some of these games are not very recent, but still very popular within specific gaming communities – for instance, *Super Mario Bros.* and the speedrunning community, or the original NES *Tetris* and the competitive community established with the *Classic Tetris World Championship* –, so that collecting data about them still makes sense.

Next, the games that are, on average, the most streamed on Twitch were identified by running a Python script that relies on the Twitch APIs in order to download data about every streamed game on the platform. After running the script for 48 hours and collecting data every 3 minutes, a list of 25 games was isolated (excluding games found in the previous step). For instance, the list contains the following titles:

- *Apex Legends*;
- *Call of Duty: Black Ops IV*;
- *Counter Strike: Global Offensive*;
- *Dota 2*;
- *Fifa 19*;
- *Fortnite*;
- *Hearhstone*;
- *League of Legends*;
- *Overwatch*;
- *PLAYERUNKOWN'S BATTLEGROUNDS*;
- *World of Warcraft*.

After the identification of the list of games to monitor, the collection of the desired data from Twitch and Twitter was carried out as described in the following subsections.

1.2 Collecting Twitch Data

The data from Twitch has been collected by running a Python script, called *twitch_collect_schedule.py*, that sends a number of requests through the Twitch API v5 every 3 minutes to obtain the data about all the streamed games, sorted by popularity [4].

The raw data collected with each request consists of a document with two fields, *total* and *top*. The value of the first field is the total number of streamed games, while that of the second is an array of documents, one for each streamed game.

From this data a new document was created consisting of two fields, *timestamp* and *data*, to store the downloaded data. The value of the first field is the time of the request, while that of the second field is the complete list of game-related documents, where some redundant fields were removed, some inner documents were flattened, and the field *game_norm_name* was added (its value was useful to integrate the data from VGChartz). Depending on some options specified when starting the script, the data can be saved locally on a json file or sent to a MongoDB collection. Figure 1 shows the structure of the Twitch data.

1.3 Collecting Twitter Data

The data from Twitter has been collected using the Python script *download_top_50_tweets.py* that downloads tweets in the time range specified in *download_top_50_games.json* configuration file. For our use case the following search parameters were used:

- At most 200 tweets for each game are collected in the given timerange.
- The eventual retweets of a given tweet are not included, as it is more interesting to monitor a heterogeneous dataset.

```
{"timestamp": "2019-06-21 16:03:58.095231",
 "data": [ {"channels": 7860,
            "game__id": 33214,
            "game_box_large": "https://...",
            "game_giantbomb_id": 37030,
            "game_logo_large": "https://...",
            "game_name": "Fortnite",
            "game_norm_name": "fortnite",
            "game_popularity": 228843,
            "viewers": 250659},

            {"channels": 2936,
            "game__id": 21779,
            ... },

            ...
      ]
}
```


Figure 1. Structure of the data collected from Twitch


```
   {
      "query": "fortnite",
      "text": "...splain dat #fortnite...",
      "language": "en",
      "date": ISODate("2019-06-14T23:59:23Z"),
      "username": "{Fang} LK"",
      "user_followers": 72,
      "user_location": "Georgia, USA",
      "retweets": 0,
      "likes": 4
   }
```


Figure 2. Structure of the data collected from Twitter

- The start date and end date can be freely chosen, even though in this project a one-day time range was chosen.
- The results are ordered according to the "mixed" criterion. This allows to collect the *n* tweets that are considered popular and *200 - n* tweets ordered by decreasing timestamp value. This was done because for popular games such as *Fortnite*, *Fifa 19* and *League of Legends* during peak hours each 3-4 seconds a tweet is published, therefore it would not have been feasible to collect every tweet for all 50 games for multiple days.
- Each game is queried using its "normalized" name: punctuation characters are removed and all letters are lowercased.

For collecting these data, the use of *Tweepy's Cursor* object [5] represented the best compromise: it allows to query tweets that were published up to 2 weeks before without the need to establish a proper streaming pipeline that would have given less control on the number of tweets collected for each day.

As it can be seen in Figure 2, not all the information returned by the Twitter API was kept, but only the most relevant fields such as the number of likes and retweets of a given tweet.

2. Data Management

MongoDB was chosen for storing all the previously described data. The following are some of the reasons that led to this choice [6]:

- High performance: with more than 5 GB of data stored and a streaming service that every 3 minutes writes around two thousands record it is important that the DB is able to execute both write and read operations in a reasonable time;
- Document oriented architecture, which makes MongoDB the natural choice for storing the JSON data collected by the two APIs;
- High availability and high scalability: even tough the data volume is not that high, if the scripts are kept active in few weeks it will be necessary to scale the DB and having the possibility to do it in a horizontal way without much effort is obviously a nice feature to have;
- Flexibility: field addition/deletion have less or no impact on the application. Twitch and Twitter APIs are expected to change every now and then, and in the future there may arise the need for collecting new fields (especially on Twitter, where a preliminary filter was performed). Therefore, having the opportunity to change data structures on the fly without disruptions and not having to modify previous records is essential for our use case.

In particular all the data are stored on a single database, called *dm_project*, which has three distinct collections (*twitter*, *twitch* and *vgchartz*). Although one of MongoDB's main characteristics is the possibility to store documents having different structure in the same collection, the little gains on overhead sizes are not worth the increase in complexity and readability [7]. At the time of writing this report, the size of the database has exceeded 5 GB.

3. Data Exploration

The collected data was explored by querying the MongoDB collections. In the following subsections an overview is given of what kind of queries can be exectuted on the data. More examples can be found in the Jupyter Notebook *03-Exploratory_queries.ipynb*.

3.1 Querying Twitch Data

Since the quantitative data available for each game consists of the number of *channels*, the number of *viewers* and the *popularity*, aggregate queries revolving around these fields were performed, such as the the average daily/hourly number of channels, average number of viewers and average popularity for each game.

Through these queries it is possible to find out which were the top streamed games on a given day/hour, or to find out, for instance, the rank of a given game, by number of

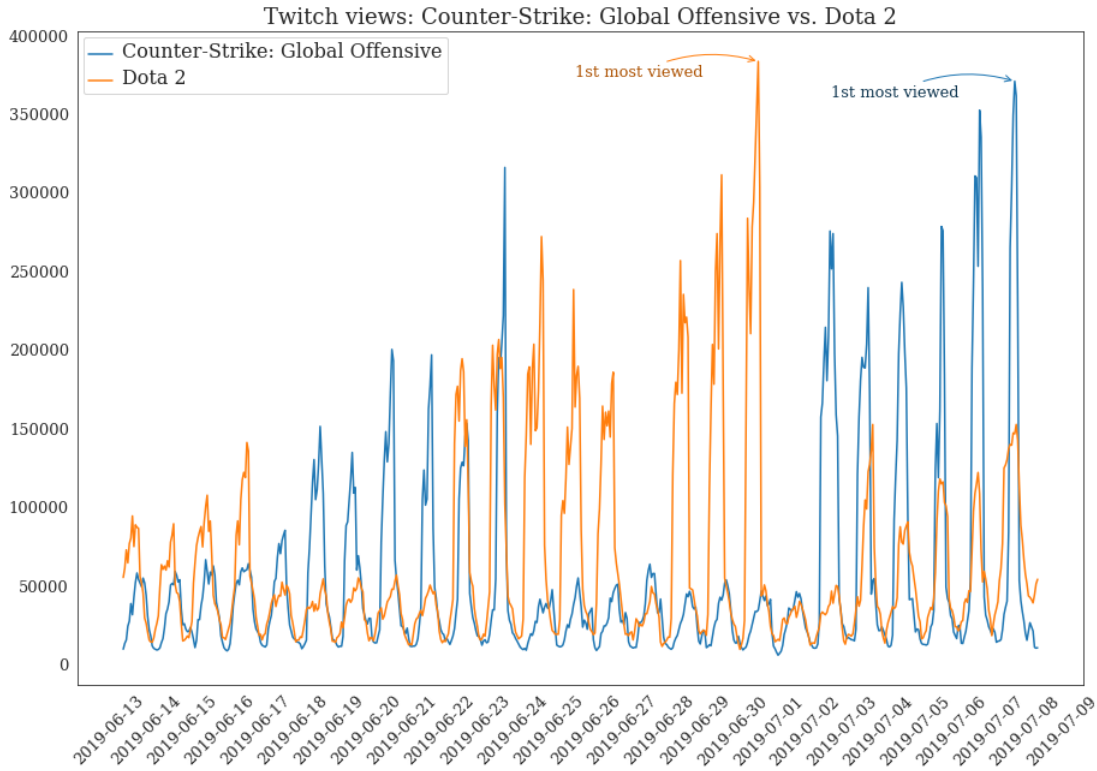

Figure 3. The viewership of Counter-Strike: Global Offensive against Dota 2

viewers, among all streamed games on a given day/hour. In this way it is also possible to understand, for each game, what are the hours and days with highest average viewers. This will become more evident by visualizing the data: for instance, by plotting the hourly average viewers of *Super Smash Bros. Melee* it is evident that, while the average views are not notable, this game can achieve very high peak viewers (usually during tournament finals), so that it can arrive at being the 6th most spectated game on Twitch.

3.2 Querying Twitter Data

Regarding tweets, the focus was given on retweets and likes values associated with tweets about the selected games. *Fortnite* is the game that usually receives more retweets on a single day: the only exceptions were *Pokémon Gold/Silver*, *Overwatch* and *Slots*, even though this last keyword is not uniquely associated with the game. The same can be said about likes, although according to this metric *Minecraft*, *Super Smash Bros. Melee* and *Pokémon Gold/Silver* are the only games that obtained better results than *Fortnite*.

Because other fields were stored, in the future it will be possible to perform other analyses and studies on Twitter data, such as:

- Sentiment analysis on tweets, for better understanding the type of reactions associated with each game.
- Location of the accounts: are the discussions about games localized in particular markets or are they spread globally?
- Join other sources for identifying "anomalies": it has been observed that *Fortnite* is usually the game with the highest retweets and likes, with some exceptions. The most plausible reason is that on some days there are particular announcements or keynotes that alter the *status quo*: for investigating this hypothesis it will be useful to check other sources such as articles posted on the most famous gaming blogs.

3.3 Measuring the "buzz"

The main goal of this project is to measure the popularity of the selected games on Twitch and Twitter. For this reason a summary metric that takes into account the popularity of each game in these two social media was created: the *buzz* value therefore corresponds to the weighted sum of the standardized values of Twitch average views, Twitter likes and retweets related to each game on a given day.

Thanks to this new feature it is possible to compute, for instance, the games with the highest and lowers buzz on each day, which game has reached the highest buzz and which the lowest, or which is the game with the highest (or lowest) total buzz in a given time range. A few examples follow.

- The highest buzz was reached by *Minecraft* on June 27, with a buzz of 6.8, while *Mario Kart DS* was the game

that generated the least interest on Twitch and Twitter, with a buzz of -0.4 on June 13.

- *Fortnite* is also the game with the highest total buzz value of 47.4, while *Call of Duty: Modern Warfare 3* is the worst performing, with a value of -5.75.

4. Data Visualization

Figure 3 shows an example of a possible visualization of the Twitch data[1].

Conclusion

For this project data from Twitch and Twitter was collected in order to monitor the popularity of a selected list of videogames on these platform. To this end, the data was stored on MongoDB and explored by querying the database and defining a new variable, *buzz*, that quantifies the popularity of a game.

An analysis of this kind can be useful in order to find out which games are more popular and during which hours or circumstances, so that a company could decide, for instance:

- when to insert ads on twitch, and on which streams;
- which games to sponsor and during which events;
- which Twitter or gaming personalities to sponsor or hire in its e-sports team;
- which games should be considered when hosting an e-sports tournament for maximising the number of visitors and partecipants;
- which games references to make during advertsing campaigns in order to attract the attention of a particular niche.

This analysis could be extended by considering more games or periodically changing the list of selected games, in order to be up to date with new releases. The Twitch data could also be expanded by collecting data about streams, so that it could be possible to identify which are the most popular streamers for each game.

Another possible development of this work is the possibility to use other data sources in order to have a broader picture of the gaming movement. In particular the two sources that would be ideal to use, together with Twitch, Twitter and VGChartz, are *YouTube* and *Reddit*. The first one is the most popular video streaming platform in the world and a valid competitor of Twitch in the gaming sector. The reason is that together with streaming channels, users can watch news, reviews and gameplay published by traditional media, specialized game websites and popular YouTubers such as *PewDiePie*, who is followed by more than 96 million people. The second one is a trending social media, where many strong and specialized communities exist, many of them having millions users, like *r/gaming*, *r/leagueoflegends* and *r/PS4* just to cite a few. In these places users share and discuss the main news of the day, in a similar way to *Twitter*, without having any limits on the number of characters.

[1]Please note that this is not the visualization that will be submitted for the Data Visualization part of the project

References

[1] VGChartz (June 2019). *About VGChartz*. Retrieved from http://www.vgchartz.com/about.php.

[2] GitHub (June 2019. *vgchartzScrape*. Retrieved from https://github.com/GregorUT/vgchartzScrape.

[3] GitHub (June 2019). *Pull Request - Improved script for downloading all games data without incurring in errors* . Retrieved from https://github.com/GregorUT/vgchartzScrape/pull/5.

[4] Twitch Api v5 (June 2019). *Games Reference*. Retrieved from https://dev.twitch.tv/docs/v5/reference/games/.

[5] Tweepy Documentation (June 2019). *Cursors tutorial*. Retrieved from http://docs.tweepy.org/en/v3.7.0/cursor_tutorial.html.

[6] DZone.com (June 2019). *When to Use (and Not to Use) MongoDB*. Retrieved from https://dzone.com/articles/why-mongodb.

[7] MongoDB documentation (June 2019). *Operational Factors and Data Models*. Retried from https://docs.mongodb.com/manual/core/data-model-operations/#large-number-of-collections.