Twitter API v2

Twitter is composed of posts called Tweets. An example of a Tweet is given in figure 1.

Figure 1: Tweet example

All the Tweets are posted by a user and their minimal structure is a text box, with a limit of 280 characters.
Through the API it is possible to ask about all the information of a post and the basic data that could be obtained are the id of a Tweet and its text. The additional information are optional fields so, if we want to add them in the response, it's necessary to specify which additional data we want in addition to the text, for example the author, attached media, interactions, metrics, geolocation or contextual information.
When additional data is added in the response only the id of that field is returned. To get the additional fields is necessary to do an expansion of those data specifying all the fields of the new object that have to be in the response.
There are two possible GET requests to retrieve the data: the first one is by the id of the post and the second one, that consent to get more specific Tweets, could be done with a query. Since the id could be unknown the query allows us to get some Tweet's id using some filters based on the content of the Tweet. For example we can ask for the tweets that contain some particular keywords, phrases or emoji, or came from a specific author.
As well as the complete tweet also the count of those matching tweets can be obtained with a GET request.

There is a rate limit imposed by the API for the number of the request that could be done in a determinate time interval (usually 15-minutes). Those limits can change based on the type of access token used, a user token or an app token.

The text inside of a Tweet can contain some different entities, the ones that are highlighted in a light blue color. Those could be hashtags[1], tags[2] to other user profiles, cashtags[3], and links to other web pages. All of these elements are links to other places in Twitter for

[1] The words that starts with # and it's a temaic aggregator
[2] The words that starts with @ and identify a user
[3] The words that starts with $ and it's like an hashtag but related to stock or cryptocurrency

example if a hashtag is clicked all the posts that contain that hashtag are shown and if a user tag is clicked the profile page of that user is opened.
The additional information we can obtain about them are their content and the start and end position in the text.
The query filter allows us to search posts that have or don't have those entities or contain some specific keywords, hashtags, tags, url or emoji.

Beside the text a Tweet could contain attachments, those could be media such as photos, videos or gifs and could be also polls, where the users can vote through different options.
Also the information about those media could be retrieved by the API and inserted in the query parameters.
To get the information about the media all the fields that have to be in the response need to be specified, and what we can obtain is the basic information about the media like the type, the dimensions or the duration but also some statistics like the number of visualizations. Only those types of data can be obtained but it is not possible to download it, more information about the content, only for images, could be obtained by the alternative text[4].
The media data could be used also for the composition of the search query asking for Tweets that have or not images, video or any type of media[5].

Another type of data that the Twitter user can insert in his Tweet is the geolocation. From it we can obtain the coordinates and a place object that contains additional information like the country, the name of the place or the type of the information represented by the place such as city or point of interest. This is not the position where the post was made and published because the user can tag whatever position he wants, to be sure of the position where the post was made it is possible to compare the position tagged in the profile of the user that posted the Tweet with the one present in the Tweet.

There is also some information that is not visible in the tweet but is useful to understand the contextual information about the tweet itself, called annotations. Those are automatically generated by the Twitter algorithm in two different ways.
The entity annotations are produced in a more programmatically way that maps a certain set of words in a set of categories: people, places, products and organizations. Each one has a score that indicates the confidence and the position of them inside the text.
The context annotations are inferred based on the Tweet text and assigned to a list of 50+ domains to categorize Tweets (Figure 2).

[4] Textual description of the image up to 1000 character long
[5] Only the media that are directly hosted by Twitter can be used to filter the search, the links to external media are not recognized as media.

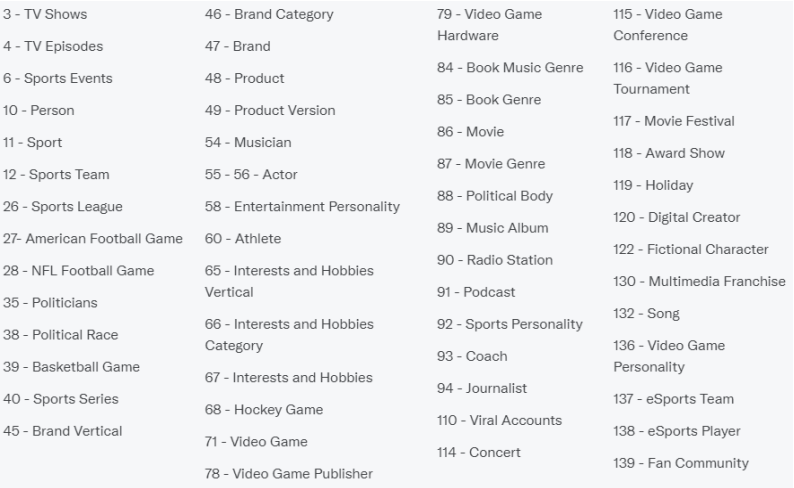

Figure 2: Context annotations domains

The search method allows you to get the Tweet from the archive but there is a more "dynamic" way to get the posts. The API gives the possibility to access the continuous Tweet stream and have real time data. There are two ways to retrieve information from that stream. The first one, the sampled stream, allows you to get a set of new random Tweets of random users, indeed the second one, called filtered stream, is more powerful because it can apply some filter, like a query, and get in real time all the Tweets that satisfy the rules.

A user can interact with a Tweet in different ways: reply[6], retweet[7] like and share (Figure 3). All the statistics of all the interactions are called engagement and we can ask to add those metrics in the response of our GET request. In addition to those metrics it is possible to make another type of request and ask for the list of all the users that liked or retweeted the post.
There are also private metrics that generate the private engagement such as the total number of impressions[8], URL and profile link clicks. Those private metrics are accessible from Tweets of owned or authorized[9] accounts.

Figure 3: User's possible interaction with the Tweet

When a user makes a reply or a retweet a conversation begins and is possible to reconstruct the conversation with some fields of the Tweet object. The id of the first post that started the conversation becomes the conversation id and it is present in every Tweet that is also a retweet and in every reply.
Also the list of all the Tweets that the searched Tweet refers to can be added in the response. The list contains the information about the type of interaction made reply, retweet or quoted retweet[10] and the id of the parent Tweet.

---

[6] A reply is like a comment in the Tweet
[7] A retweet is like a repost with the possibility to add ulterior text of max 140 characters
[8] Number of visualization of the Tweet
[9] Account that has authorized the developer app to access by granting it access to that account
[10] Retweet with comments

Users

For every user in Twitter there is a profile page (Figure 4) and inside of it different information can be found.

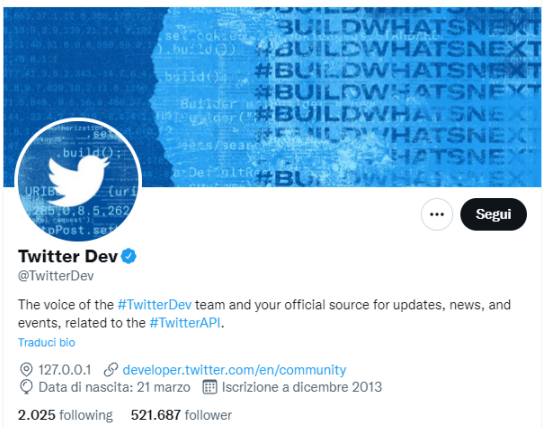

Figure 4: User profile example

Every user has a name and a username[11]. These two information could be used to search the users through a GET request. The basic data that is presented in the default response are the id, the name and the username.

In each user profile could be present a textual description that could contain different fields like URLs, hashtags, mentions and cashtags with the same structure of the entities that could be found in the Tweet object.

Additional information of the user can be retrieved like the date of creation of the account, the location, the profile verification[12] and the profile image URL. In the response can be added only information about the account but not any personal information like age or gender.

Also the users have metrics that show the number of followers and following profiles. In the Twitter website when the follower or following numbers are clicked a new page that contains the complete list of users that the user follows or is followed by the user is shown. Also with the API using a different request it is possible to retrieve those lists.

In the profile all the Tweets that a user posted are present, this is called timeline, and it is possible, when the id of a user is known, to explore that timeline and get the list of those Tweets.

Historical API

With the standard usage of the API only the most recent seven days from the Tweet archive can be obtained. There are also two more methods that could be used to get access to all the Tweets starting from the first Tweet posted. Those two methods called historical power-track and Full-archive search, examine all the Tweets during the time of interest but are based on significantly different architectures.

---

[11] Unique identifier of a Twitter profile that starts with a @
[12] A blue tick near the name

Historical Power-Track (HPT) is built on top of a file-based archive and to examine those and uses a batch, job-based design where the API is used to move a job through multiple phases. Each Power-Track job consists of the following steps: after the creation of a new job the API generates a ballpark estimate[13] of the expected data volumes and time required to complete the job, with another request is possible to accept or reject the job. When the job is finished it is possible to get the URLs of the data[14] that is used to download the data files.
The results are delivered in files with at least one Tweet in 10-minute frequency, the dimension of the file is variable and depends on the number of Tweets found in the 10-minute time window (if no Tweet is found the empty page is not received).
To filter the results it is possible to add in the request up to 1000 rules. A rule is like a query with a maximum length of 2048 characters, that supports different operators that could be combined with logical operators.

Full-Archive Search (FAS) delivers matching Tweets paginated in a small number of Tweets per page. This search is provided with both versions of the API, v1.1 and v2.0.
The first is based on the RESTful request/response pattern where a single PowerTrack rule is submitted for a request.
The 2.0 endpoint instead works with the same request syntax of a Tweet recent search.
In both of the cases a response page that contains up to 500 results and a next page token. Since the results are paginated all the result pages need to be stitched together to get the complete dataset.

There are many differences (Table 1) between those two methods: the number of rules are 1000 for the HPT and only one for the FAS but the set of applicable operators is the same for both the methods but there is a subset of operators that are not supported by FAS (Figure 5).

- bio:	- profile_point_radius:	- friends_count:
- bio_location:	- source:	- has:lang
- profile_bounding_box:	- is_quote:	- retweet_of_status_id:
- sample:	- follower_counts:	- url_contains:
- bio_name:	- profile_subregion:	- in_reply_to_status_id:
- contains:	- url_titile:	- listed_count:
- statusess_count:	- url_description:	

Figure 5: Operators not supported by FAS

HPT generates a time series of data files, one for each 10-minute job, indeed the FAS generates data arranged in a results array in a JSON format for each response; FAS provide a count endpoint that is used to generate minutely/hourly/daily time-series of matching tweets and the FAS provide an order of magnitude estimate for the number of Tweet will match.

[13] An estimate of 100 could represent 0-999 tweets, an estimate of 100,000 could represent 100,000-999,999 tweets
[14] Data files generated are available for 15 days

	HPT	FAS v1.1	FAS v2.0
Response type	Files to download	JSON array	
JSON type	v1.1 response		v2.0 response
Response limits	No limitations	500 per response	
Rule type	PowerTrack rule		API v2.0 query
Rules number	up to 1000 for a job	1 for request	
Research time[15]	6 hours	> 6 hours	> 6 hours
Count type	estimated	precise	count method
Rate limits	- 60 jobs for day - 30 jobs for hour - 2 job estimation concurrently - 2 job running concurrently	- 180 for 15-min (user auth[16]) - 450 for 15-min (app auth[17])	- 300 for 15-min - 1 for 1-sec

Table 1: Historical API comparison

There are different reasons to choose one method instead of another: for an high volume dataset (not less than few millions) the HPT is recommended because with the FAS the research time is bigger due to the high number of request to make, HPT is also better for use cases where there are less time sensitive researches and have a large query ruleset. If the goal is to count the data FAS is a better choice because it provides a more accurate estimation and is better also for applications that require near-instant results because FAS delivers quick responses of small dimensions.

[15] In a research for 8 millions tweets
[16] If a user authentication token is used in the request
[17] If an app authentication token is used in the request