Twitter API report

Prepared for

Mr. Song

By

Jiahao Fang

Oct 23, 2022

Contents

Overview

As one of the world's biggest social network platforms, Twitter hosts abundant user-generated posts, which closely reflect the public's reactions. Twitter also provided strong dev API for academic usage, it allows developers to access to Tweets, Direct Messages, Spaces, Lists, users, and more. For the highest-level access, the user has the access to full-archive search and full-archive Tweet counts, which could be a useful tool for sentiment analysis.

The report will mainly focus on three parts:

The first part is about the possible usage of Twitter API

The second part is the introduction of each passages including its functionality, cost and API capabilities.

The third part is the implementation of each function

Project scope

For sentiment analysis, we mainly focusing on two functions of API:

1. Full Archive Tweets search and Tweets counts
    a. By analyzing the tweets overall sentiment over a period, we can potentially generate trading signal for
    b. Twitter API could return tweets including its contents, users, like counts, retweet counts and reply counts, which could be possible importance weights for each tweet to better evaluate public reactions.
2. Tweets streaming
    a. After analyzing the historical data and integrate corresponding features into model, twitter API allows user to stream tweets asynchronously in real time with certain filtering rules like keywords (crypto) or users (Elon_musk).

API introduction

Twitter API

Official API that lets you access and interact with public Twitter data. User can use the Twitter Streaming API to connect to Twitter data streams and gather tweets containing keywords, brand mentions, and hashtags, or collect tweets from specific users.

The twitter API has three versions running concurrently, **Twitter API v2, Twitter API v1.1** and **Enterprise – Gnip 2.0**. Here we will only discuss Twitter API v2 and Enterprise – Gnip 2.0.

Twitter API v2 has powerful functions and faster response time than Twitter API v1.1. Twitter API developers strongly recommended all the developer to migrating to v2 of the API.

There are three access levels for V2 API, I listed the most important features that related to our project:

Level	Essential	Elevated	Academic Research
Cost	Free	Free	Free
Application requirement	basic	Fill out a form about the intention of using API	Researcher only, require applicants academic profile including academic institution and google scholar profile or lab website
Recent tweets search	Yes	Yes	Yes
full-archive search	No	250 requests for 30-days search and 50 requests for full archive (Sandbox level access from V1.1)	Yes
Request limitation	500k Tweets per month	2 million Tweets per month	10 million Tweets per month
Search rate limit (recent)	450 requests / 15 mins 100 results per response	Same	Same
Search rate limit (Full archive)	No	30 requests/ min 100 results per request	300 requests / 15 mins 500 results per response
Stream rate limit	25 requests per 15 minutes	50 requests per 15 minutes	100 requests per 15 minutes
V1.1 and enterprise endpoint access	No	Yes	Yes
Upgrade	No	Yes	Yes

The 30 days and full-archive search function is still under old version endpoint. Thus, only with the access of V1.1 and enterprise endpoint could access the full archive search.

By obtaining the elevated level access, we can upgrade our sandbox level full archive search to premium full archive tweets search and count (500 results per request& 60 requests/min& 1.25M monthly Tweet cap), Here are the different prices of level of usage:

Total request/ Per month	Price / $ Per month
100	99
250	224
500	399
1000	774
2500	1899

Enterprise API provide the highest level of access and customized data rates according to user's usage needs. It has the access to the full archive of public Tweets and real time streaming API with advanced filtering tools.

From its website, I couldn't find more useful information if I don't apply for the enterprise access, I assume it is only for commercial usage so the function and limit of API is highly customized and there will be support team contact the applicant for further customization.

Twitter API Link: https://developer.twitter.com/en/docs/twitter-api

Enterprise API document: https://developer.twitter.com/en/docs/twitter-api/enterprise

Enterprise API access application: https://developer.twitter.com/en/products/twitter-api/enterprise/application

Tweepy

Tweepy is an easy-to-use Python library for accessing the Twitter API. Developer could easily implement the major twitter API functionality by using tweepy. It means this package still **needs the access and authentication of twitter API**. In the code implementation part, I also use tweepy to demonstrate streaming function.

Link: https://docs.tweepy.org/en/stable/

Snscrape

snscrape is a scraper for social networking services (SNS). For Twitter, it scrapes users id, user profiles, hashtags, searches, tweets (single or surrounding thread), list posts, and trends **without speed and volume limitation and no authentication needed**. In the code implementation,

Link: https://github.com/JustAnotherArchivist/snscrape

Code implementation

Snscrape_search:

This code can search and store the tweets contain certain keywords within certain period. It can be accomplished because the twitter's advanced search tool allows users to search tweets by date. The limit of this function is that we cannot narrow down the search period shorter than one day like search the tweets in 2 hours period.

Thus, this code is designed to scrape the tweets using multiple threads, the function will split the user search date into multiple shorter time periods and each period will have the corresponding thread to implement the search. However, due to the limitation of twitter search tool, **each thread will be at least assigned a whole day**, when the keyword is rather simple, like 'crypto', then everyday will have tens of thousands of tweets contains that keyword. For example, keywords 'crypto' is very popular nowadays, when I implemented the function with keyword 'crypto', it took 5 minutes to scraper first half hour data of a day. So, user should be aware that if we want to download the tweets with popular keyword, no matter how many threads we use, it will take hours to scraper the whole day's data for each thread.

Speed_test:

This code is used for testing the efficiency of multithread. It contains two parts.

The first test is for testing how many times for program to scrape 10000 tweets for 120 threads

Test result: 4.8 seconds

The Second test if for testing how many times for program to scrape 1000 tweets for 1 thread and 100 threads

Test result: 29 seconds for 1 thread, 1.2 second for 100 threads

Tweepy_stream:

This code is only for demonstration purpose. Run the code, it will stream the tweets and print in the terminal with filtered keywords. It still has many problems like the streaming connection will disconnect in 5 minutes. Thus, this function is still under development.