{TWITTER DATA COLLECTION USING TWARC}

Twitter Data Collection Using Twarc

Archiving Twitter JSON data

Researchers can use Twitter data to inform their work. This kind of data is not limited to a single discipline, and can support questions from both the social and engineering sciences. With Twitter data you can create a [network analysis](#), which can help increase understanding of how movements form and organize online. You can also do things like conducting a [sentiment analysis](#) on a dataset, which applies Natural Language Processing (NLP) techniques to determine Twitter users' attitudes towards a subject. This tutorial explains how to collect and ethically share Twitter JSON data using an open source tool, [Twarc](#).

Difficulty level: Beginner

Optimized for: Mac users. Windows users can view the tutorial [here](#).

Prerequisite(s)
- [Tweet JSON](#)
- [Command Line](#)
- [Collection Design](#)

Tutorial Key
- `Command Line arguments will be displayed in this format`
- 🎉The party popper emoji signals the end of each set of instructions 🎉

Lesson objectives
- Use Twarc's *search* command to collect Twitter JSON data
- Collect the unique Twitter ids from a dataset using Twarc's *dehydrate* command
- Learn how to ['rehydrate'](#) a list of Twitter ids using the *hydrate* command

Key Terms
- Terminal - OS X Command Line
    - A text interface for your computer. Terminal receives commands, and then passes those commands on to the computer's operating system to run.
- Twarc
    - A command line tool and python library

- Python
  - The programming language that Twarc is developed in
- Application Programming Interface (API)
  - The interface that allows software applications to communicate with one another
- JSON - JavaScript Object Notation
  - A minimal, human-readable format for structuring data. Twitter data is in JSON format.
- `Dehydrate` - Twarc Command
  - Generates an id list from a file of tweets
- `Hydrate` - Twarc Command
  - Reads a file of tweet identifiers and write out the tweet JSON for them using Twitter's status/lookup API.

Table of Contents

Introduction

Twarc was created by the team behind [Documenting the Now](), a project aimed at promoting ethical collection and archival processes for social media data. This tutorial will walk you through how to install Twarc, and then use it to collect Twitter data based off a set of search terms (queries). You will then learn how to ethically share your collection using Twarc's *dehydrate* and *hydrate* commands. At the end of tutorial, a list of additional commands for Twarc are available.

Register an application

Before using Twarc you will need to apply for a Twitter developer account and then register an application. Follow the steps below to register for a Twitter developer account.

Step 1: Apply for a developer account
1. You can apply for a developer account [here]().
   Note: You must have a Twitter account to apply for a Developer account
2. Select 'I am requesting access for my own personal use'
3. Select the use cases you are interested in (Ex: academic research)
4. In the 'Describe in your own words what you are building' section, provide the following information:
   a. Use case: Describe your research area (Ex: Analyzing tweets using the hashtag #BlackLivesMatter to further research on how social justice movements organize online)
   b. Analysis: Provide further details on your use case while keeping with the Twitter's [API Terms of Service]().
   c. Content: Describe whether or not you will be sharing content. Keep in mind that Twitter has strict privacy guidelines to protect its users.
   d. Display: Describe how your results will be shared. (Ex: Academic paper with all identifiable information removed from the content shared)
   Note: When applying for developer access, make sure to keep Twitter's [API Terms of Service]() in mind. Twitter wants to ensure that you will protect the privacy of their users.

5. You will receive an email once your request has either been approved or denied by Twitter.

Step 2: Register an application
1. Register an application here.
2. Once your application has been registered, write down the following:
   a. The consumer key
   b. Consumer secret
   c. Click to generate and then write down the access token and access token secret

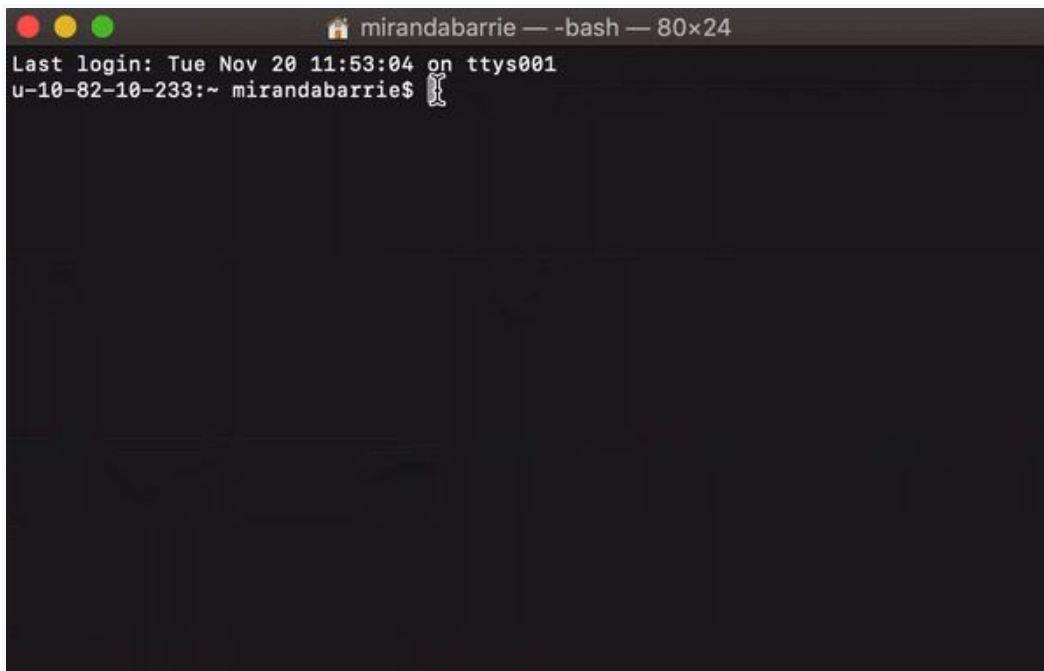Once you have completed steps 1 and 2, you can install Python and Twarc.

Install Python and Twarc

1. Download and install the latest version of Python here.
2. Open the Terminal application (Located in the applications folder)

3. Pip install Twarc by entering the following command:

```
pip install twarc
```


Note: *Pip* is already installed if you are using Python 2 ≥ 2.7.9 or Python 3 ≥ 3.4

🎉Congratulations! You've installed Twarc!🎉

Configure Twarc

To get started, you will need to tell Twarc about your application API keys and grant access to one or more Twitter accounts. Follow these directions to configure Twarc:

1. Enter the following command in Terminal:

   `twarc configure`

2. Twarc will ask you to enter several keys. You should have these keys ready to go after [registering an application](#).
3. Enter the required access keys to configure Twarc

   🎉Great job! You've configured Twarc!🎉

Use Twarc to collect tweets

Now that you've configured Twarc, you can begin collecting Tweets! Follow the instructions below to collect a set of tweets. For this tutorial, you will be using Twarc's *search* command to return tweets containing 'BlackLivesMatter' occurring within the past 7 days. An explanation of the search command as well as a list of [additional commands](#) is provided at the end of this tutorial.

1. Create a new folder on your desktop titled 'BlackLivesMatter_Tweets'.
2. Open the Terminal application
3. Change directories by starting the command `cd` and then dragging your 'BlackLivesMatter_Tweet' folder into Terminal. Hit return to complete the command once you have dragged the folder into terminal successfully.
   *Tip: Make sure to leave a space between the command **cd** and the filepath.*
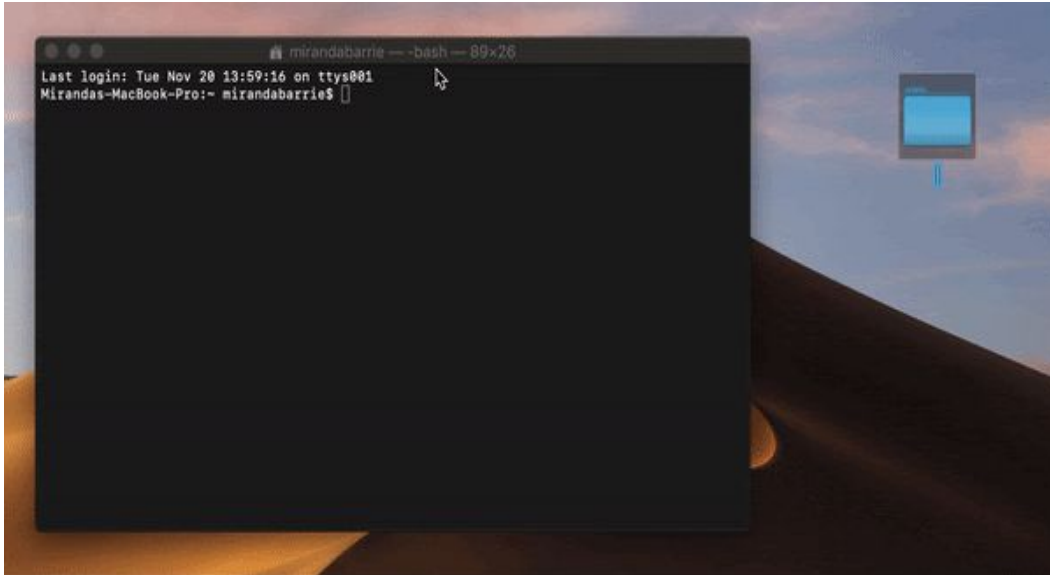
4. Make sure you are in the right directory by entering the command `pwd`. You should be in your 'BlackLivesMatter_Tweets' directory.
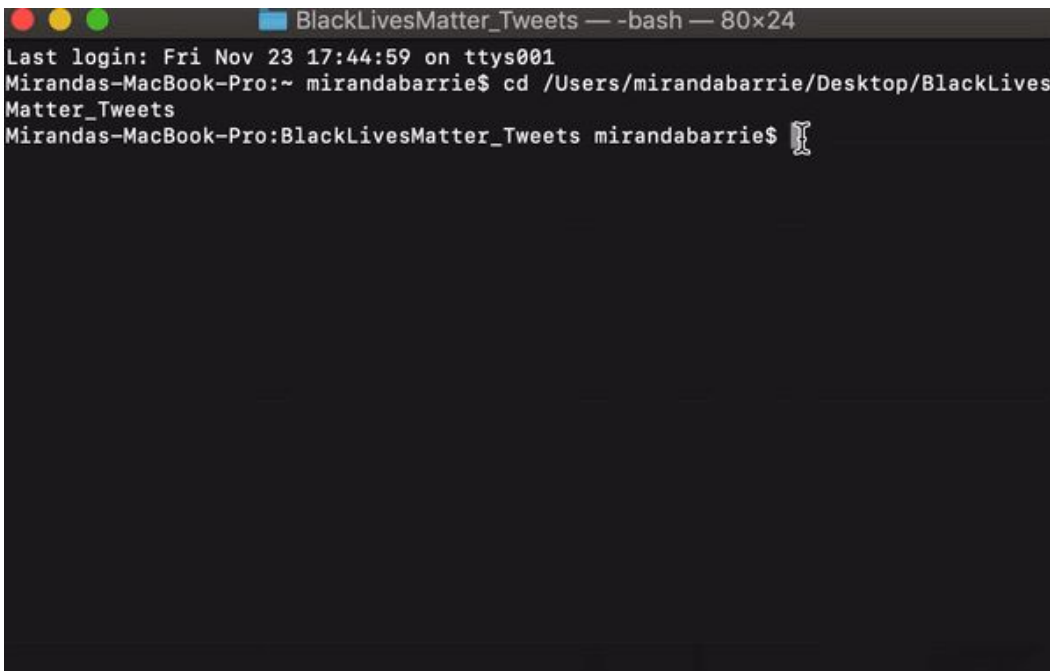
5. Now that you are in the right directory, enter the following command to start collecting tweets:

```
twarc search blacklivesmatter > blacklivesmatter_tweets.jsonl
```

Tip: You can copy and paste these commands into Terminal to avoid errors
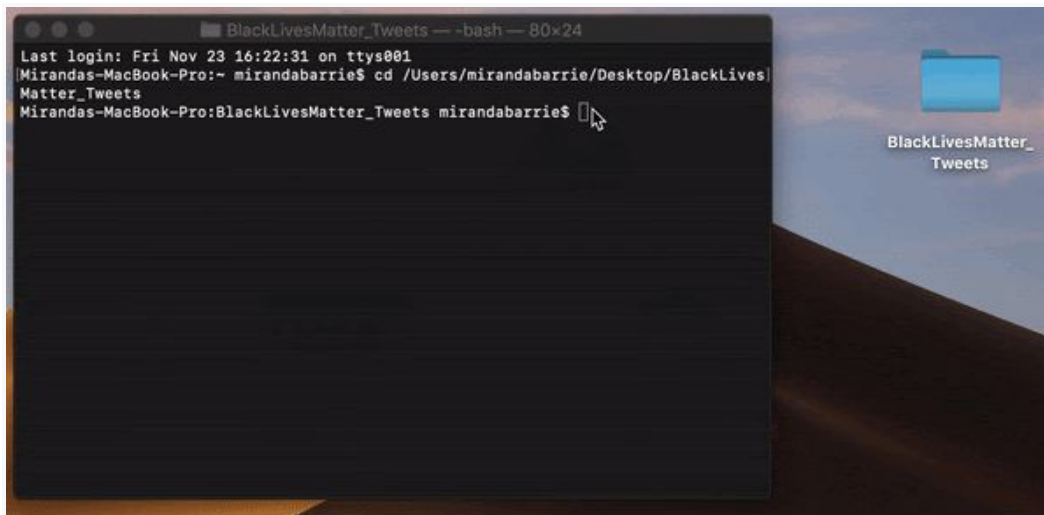
Important: Your collection may take some time to return all Tweets. You can tell when the process is complete when it returns to the shell ($) prompt. Follow step number 6 to ensure your collection has started properly.

6. You can check to make sure your command was successful by clicking on your 'BlackLivesMatter_Tweets' folder. Inside you should see your 'blacklivesmatter_tweets.jsonl' file and a 'twarc.log' file.
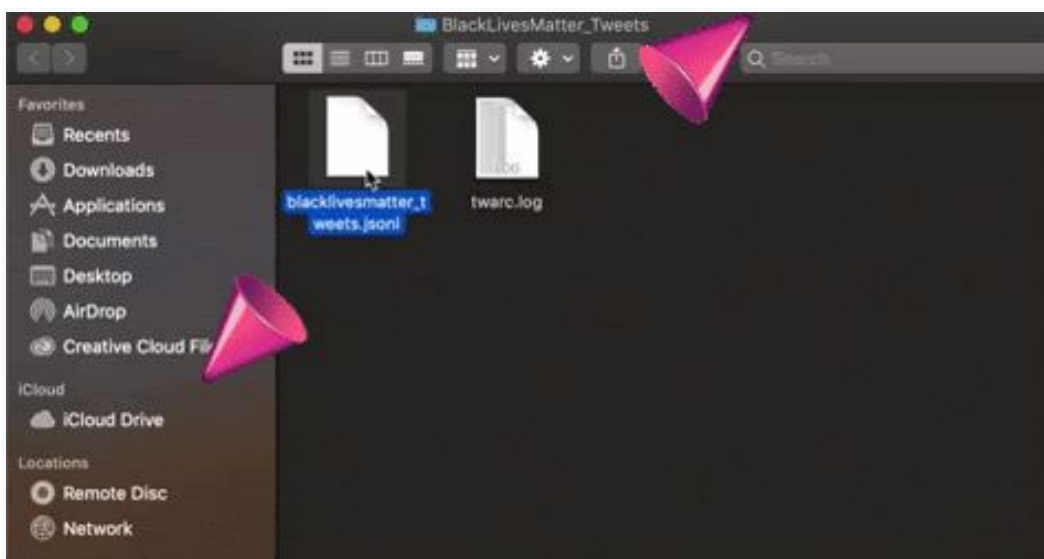
7. For the purposes of this tutorial, you can cut the search short. After entering the initial `search` command, wait **5 minutes** and then enter Ctrl + C to stop the search.
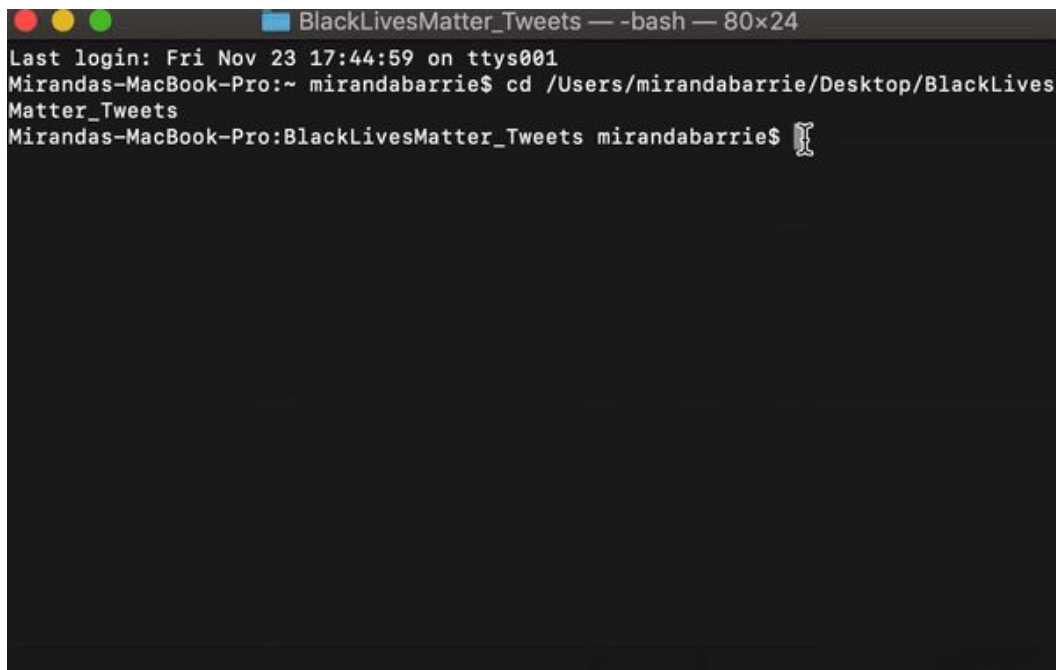
Note: If you wanted to collect the full set of tweets, you would wait until the process was finished. You can tell when a process is complete when it returns to the shell ($) prompt.

🎉Well done! Your Twitter data collection is ready to go! Your tweets are in JSON format in your 'blacklivesmatter_tweets.jsonl' file.🎉

Dehydrate your dataset

Each Tweet in your dataset has a unique identifier. Twarc's *dehydrate* command will generate a list of tweet ids from a file of tweets. You are going to dehydrate your 'blacklivesmatter_tweets.jsonl' file so that you can share your dataset while keeping to Twitter's [API Terms of Service](#).

1. Make sure you are in the 'BlackLivesMatter_Tweets' directory by entering the command `pwd`.

2. Then enter the command `ls` to list all files in the directory. You should see the files 'blacklivesmatter_tweets.jsonl' and 'twarc.log'.

3. Now you're ready to dehydrate your tweets! Enter the following command:

```
twarc_dehydrate blacklivesmatter_tweets.jsonl >
blacklivesmatter_tweet_ids.txt
```

Tip: Instead of typing out the full .jsonl file name, you can begin entering the file name and then use tab to fill the rest of the file name.

Important: Twitter's [API Terms of Service](#) discourage people from making large amounts of raw Twitter data available on the Web. The data can be used for research and archived for local use, but not shared with the world. Twitter does, however, allow files containing Tweet identifiers to be shared, like the file you just created, which can be useful when you would like to make a dataset of tweets available. You can then use Twitter's API to 'hydrate' the data, a process explained in the [next section](#), to retrieve the full JSON for each identifier. This is particularly important for the verification of social media research.

🎉You now have a text file containing the unique tweet ids of all tweets in your dataset! Your tweet ids are located in your 'BlackLivesMatter_Tweets' folder in the 'blacklivesmatter_tweet_ids.txt' file.🎉

Rehydrate your dataset

Twarc's *hydrate* command will read your file of unique identifiers and write out the tweet JSON for them using Twitter's [status/lookup API](#). This is useful if you have a set of Twitter ids from another institution and would like to view the full dataset. Documenting the Now has a collection of Tweet ids that you can explore and rehydrate [here,](#) but for this tutorial we will use the 'blacklivesmatter_tweet_ids.txt' file you created when you dehydrated your dataset.

1. Make sure you are in the 'BlackLivesMatter_Tweets' directory by entering the command `pwd`.

2. Rehydrate your dataset by entering the following command:

```
twarc hydrate blacklivesmatter_tweet_ids.txt >
blacklivesmatter_tweets_hydrated.jsonl
```

🎉Nice work! You now have your tweets in JSON format ready to go in your 'BlackLivesMatter_Tweets' folder. You can always check your folder to confirm your .jsonl file is there.🎉

Twarc commands

Now that you know how to perform a *search* command to collect Twitter data, *dehydrate* an existing dataset, and then *hydrate* a list of unique Tweet ids, you can move on to more complex commands to tailor your search to your research needs. Below you will find a list of commands that will allow you to create a more targeted collection.

Search

This uses Twitter's [search/tweets](#) to download *pre-existing* tweets matching a given query.

```
twarc search blacklivesmatter > tweets.jsonl
```

It's important to note that search will return tweets that are found within a 7 day window that Twitter's search API imposes. If this seems like a small window, it is, but you may be interested in collecting tweets as they happen using the filter and sample commands below.

The best way to get familiar with Twitter's search syntax is to experiment with [Twitter's Advanced Search](#) and copy and pasting the resulting query from the search box. For example here is a more complicated query that searches for tweets containing either the #blacklivesmatter or #blm hashtags that were sent to Deray McKesson, an activist.

```
twarc search '#blacklivesmatter OR #blm to:deray' > tweets.jsonl
```

Twitter attempts to code the language of a tweet, and you can limit your search to a particular language if you want. This example limits the search to Tweets in the French language:

```
twarc search '#blacklivesmatter' --lang fr > tweets.jsonl
```

You can also search for tweets with a given location, for example tweets mentioning *blacklivesmatter* that are 1 mile from the center of Ferguson, Missouri:

```
twarc search blacklivesmatter --geocode 38.7442,-90.3054,1mi > tweets.jsonl
```

If a search query isn't supplied when using *--geocode* you will get all tweets relevant for that location and radius:

```
twarc search --geocode 38.7442,-90.3054,1mi > tweets.jsonl
```

Filter

The filter command will use Twitter's [statuses/filter](#) API to collect tweets as they happen.

```
twarc filter blacklivesmatter,blm > tweets.jsonl
```

Please note that the syntax for the Twitter's track queries is slightly different than what queries in their search API. So please consult [the documentation](#) on how best to express the filter option you are using.

Use the following command line argument (enter the following command) if you would like to collect tweets from a given user ID as they happen. This includes retweets. For example, this will collect tweets and retweets from CNN:

```
twarc filter --follow 759251 > tweets.jsonl
```

You can also collect tweets using a bounding box. Note: the leading dash needs to be escaped in the bounding box or else it will be interpreted as a command line argument!

```
twarc filter --locations "\-74,40,-73,41" > tweets.jsonl
```

If you combine options they are OR'ed together. For example this will collect tweets that use the blacklivesmatter or blm hashtags and also tweets from user CNN:

```
twarc filter blacklivesmatter,blm --follow 759251 > tweets.jsonl
```

Sample

Use the sample command to listen to Twitter's [statuses/sample](#) API for a "random" sample of recent public statuses.

```
twarc sample > tweets.jsonl
```

Users

The users command will return user metadata for the given screen names.

```
twarc users deray,Nettaaaaaaaa > users.jsonl
```

You can also give it user ids:

```
twarc users 1232134,1413213 > users.jsonl
```

If you want you can also use a file of user ids, which can be useful if you are using the followers and friends commands below:

```
twarc users ids.txt > users.jsonl
```

Followers

The followers command will use Twitter's [follower id API](#) to collect the follower user ids for exactly one user screen name per request as specified as an argument:

```
twarc followers deray > follower_ids.txt
```

The result will include exactly one user id per line. The response order is reverse chronological, or most recent followers first.

Friends

Like the followers command, the friends command will use Twitter's [friend id API](#) to collect the friend user ids for exactly one user screen name per request as specified as an argument (command):

```
twarc friends deray > friend_ids.txt
```

Trends

The trends command lets you retrieve information from Twitter's API about trending hashtags. You need to supply a [Where On Earth](#) identifier (woeid) to indicate what trends you are interested in. For example here's how you can get the current trends for St Louis:

```
twarc trends 2486982
```

Using a woeid of 1 will return trends for the entire planet:

```
twarc trends 1
```

If you aren't sure what to use as a woeid just omit it and you will get a list of all the places for which Twitter tracks trends:

```
twarc trends
```

If you have a geo-location you can use it instead of the woeid:

```
twarc trends 39.9062,-79.4679
```

Behind the scenes twarc will lookup the location using Twitter's [trends/closest](#) API to find the nearest woeid.

Timeline

The timeline command will use Twitter's [user timeline API](#) to collect the most recent tweets posted by the user indicated by screen_name.

```
twarc timeline deray > tweets.jsonl
```

You can also look up users using a user id:

```
twarc timeline 12345 > tweets.jsonl
```

Retweets

You can get retweets for a given tweet id like so:

```
twarc retweets 824077910927691778 > retweets.jsonl
```

Replies

Unfortunately Twitter's API does not currently support getting replies to a tweet. So twarc approximates it by using the search API. Since the search API does not support getting tweets older than a week twarc can only get all the replies to a tweet that have been sent in the last week.

If you want to get the replies to a given tweet you can:

```
twarc replies 824077910927691778 > replies.jsonl
```

Using the *--recursive* option will also fetch replies to the replies as well as quotes. This can take a long time to complete for a large thread because of rate limiting by the search API.

```
twarc replies 824077910927691778 --recursive
```

Lists

To get the users that are on a list you can use the list URL with the *listmembers* command:

```
twarc listmembers https://twitter.com/edsu/lists/bots
```