The rise of the internet and mobile telecommunications has created the possibility of using large datasets to understand behavior at unprecedented levels of temporal and geographic resolution. Online social networks attract the most users, though users of these new technologies provide their data through multiple sources, e.g. call detail records, blog posts, web forums, and content aggregation sites. These data allow scholars to adjudicate between competing theories as well as develop new ones, much as the microscope facilitated the development of the germ theory of disease. Of those networks, Twitter presents an ideal combination of size, international reach, and data accessibility that make it the preferred platform in academic studies. Acquiring, cleaning, and analyzing these data, however, require new tools and processes. This Element introduces these methods to social scientists and provides scripts and examples for downloading, processing, and analyzing Twitter data.
## Contents

1 **Twitter** .......................................................... 5
   1.1 Why Twitter? .................................................. 7
   1.2 Types of Questions ......................................... 9
   1.3 A Note on Programming .................................... 10

2 **Acquire Data from Twitter** ...................................... 13
   2.1 Acquire on Your Own ........................................ 13
      2.1.1 REST API ............................................. 13
      2.1.2 Streaming API ....................................... 16
      2.1.3 Access Constraints and Replication .................. 18
   2.2 Collaborate .................................................. 21
   2.3 Purchase ..................................................... 22

3 **Process Data** .................................................... 24
   3.1 Getting Started ............................................. 24
      3.1.1 Create Account ....................................... 24
      3.1.2 Register as Application .............................. 28
   3.2 Using the REST API ......................................... 33
      3.2.1 User’s 3200 Tweets .................................. 33
      3.2.2 Tweets by ID ......................................... 34
      3.2.3 Search tweets ....................................... 35
      3.2.4 Followers ............................................ 37
      3.2.5 Following ............................................ 38
   3.3 Using the Streaming API ..................................... 39
      3.3.1 Random sample ....................................... 42
      3.3.2 Language sample ..................................... 43
      3.3.3 Geographic sample ................................... 43
      3.3.4 Specific text ......................................... 44
      3.3.5 Specific people ....................................... 45
   3.4 Converting to Local Time .................................. 46
   3.5 Improving Location ......................................... 47
   3.6 Storing Tweets ............................................... 48

4 **Clean and Analyze** .............................................. 52
   4.1 Clean ......................................................... 53
      4.1.1 Tweets From Sifter .................................. 53
      4.1.2 Tweets from REST API ............................... 54
      4.1.3 Tweets from Streaming API .......................... 55
   4.2 Text Analysis ............................................... 56
   4.3 Event Detection ............................................. 72
   4.4 Links and Imagery ......................................... 76
   4.5 Enriching Tweets .......................................... 78
5 Twitter in the Social Sciences

5.1 Research on Social Media ........................................... 80
5.2 Competitors .......................................................... 87

6 Discussion

6.1 Limitations ............................................................ 91
6.2 Ethics ................................................................. 93
6.3 Is Twitter a Social Network or Media Platform? ........ 97
Glossary

API Application Programming Interface. A standardized programming interface for accessing data or algorithms from a website.

GDELT Global Database of Events, Location, and Tone. A dataset that reads newspapers in multiple languages and codes them for political events.

GPS Global Positioning System. In this Element, “GPS coordinates” refers to the longitude and latitude that identifies the location from which a tweet was created.

HTTP Hypertext Transfer Protocol. The markup language computers use to talk to each other over the internet.

ICEWS Integrated Conflict Early Warning System. A dataset that reads newspapers in multiple languages and codes them for political events.

IRB Institutional Review Board. Bureaucracies located within universities which review any research involving human subjects.

IP Internet Protocol. “IP Address” refers to the numerical identifier for a computer when communicating over the internet.

JSON JavaScript Object Notation. A format for writing data that is easy for humans to read and compatible with data structures in most programming languages (dictionaries in Python, lists in R). It takes the form of key:value pairs that can be nested. The following example is a nested JSON object that shows names (keys) and the gender (nested key) and height (nested key) of each name: {‘Zachary’:{‘Gender’:'Male', ‘Height’:'Average'} ‘Jessica’:{‘Gender’:'Female', ‘Height’:'Average’}}. See the JSON reference page for more information.

NBA National Basketball Association. A professional sports league that figures prominently throughout examples.

NoSQL Not Structured Query Language. A family of languages for retrieving data from databases that do not use table formats.

RAM Random Access Memory. The working memory for a computer.

REST API Representational State Transfer API. A standard set of operations to allow computers to exchange static information.

SQL Structured Query Language. A language used for retrieving data from database tables.

TCAT Twitter Collection and Analysis Toolkit. A series of tools for acquiring and analyzing Twitter data without programming knowledge.

URL Uniform Resource Location. The address of a website.

UTF-8 Unicode Transformation Format. A standardized method for representing symbols, including emoji, across human languages.
1 Twitter

The increasing prevalence of digital communications technology - the internet and mobile phones - provides the possibility of analyzing human behavior in a level of detail previously unimaginable. Blogs, content aggregation sites, internet fora, online social networks, and call data records provide text, social network, and location data that vary by the second. For political scientists interested in questions about elections, language, political communication, conflict, or spatial diffusion, among others, the rise of these technologies holds much promise (Grimmer and Stewart, 2013; Bail, 2014).

These data require new tools to acquire, process, and, sometimes, analyze. These tools are no more difficult to learn and use than other qualitative and quantitative methods, but they are not commonly taught to social scientists. There also exists no canonical text, in journal or book form, that explains the strengths and weaknesses of these data and tools. This Element provides a systematic introduction to these data sources and the tools needed to benefit from them.

While digital communications technology provide data through numerous platforms, I focus on one, the social network Twitter. With over 280 million accounts creating 500 million messages per day, it is one of the largest social networks. Its data are also relatively easy to access, unlike Facebook’s. While other social media platforms and websites also facilitate data access, none are as general purpose as Twitter. Twitter’s global reach, large user base, and data openness make it the preferred platform for large-scale studies of human behavior.

I use the Twitter behavior of 21 accounts from Egypt and Bahrain as a running example. Nineteen of these accounts belong to civil society actors from four different social movements, and two belong to the Bahraini government. I detail how to purchase their tweets, download their old tweets, download specific tweets based on a unique tweet identifier, download their new tweets as they are created, and analyze these tweets. Analysis will focus on textual and spatial analysis; for an explanation of network analysis using Twitter, see Steinert-
Threlkeld (2016). Throughout these examples, I provide full working code as well as any data necessary.

The rest of this Element contains five sections. Section 2 explains where to acquire Twitter data for your research. You can download the data yourself through the REST or streaming APIs. (See the glossary for a definition of REST, API, and other acronyms.) The REST API is for finding user metadata, account relationships, and old tweets. The streaming API is for downloading data in real time. You can purchase them, or you can work with a collaborator who has obtained data that matches your research interests.

Section 3 details how to acquire Twitter data from the REST and streaming APIs. It starts with creating a Twitter account, which is necessary to access the API. It then provides code for the most common data needs: downloading a user’s tweets, downloading specific tweets, finding tweets, or downloading lists of an account’s followers and accounts a specific account follows. The section also shows how to connect to the streaming API for a random sample. The streaming API accepts parameters for language, keywords, accounts, and places, and the section shows how to use those as well. It then finishes with a discussion for different data storage approaches.

Section 4 shows how to conduct common analyses on data acquired via the methods in Sections 2 and 3. Processing is required to move from raw tweets to tweets in rectangular format. The section then shows how to conduct text analysis using tweets and detect real world events from them. It also discusses possibilities of image analysis and generating metadata for a tweet, such as demographic information.

Section 5 discusses how social scientists have used Twitter to date. Only in the last 5 years have scholars started to incorporate Twitter into quantitative analyses. Work has focused on using Twitter to measure conflict dynamics, social networks, political preferences, legislative responsiveness, economic outcomes, and mobilization for contentious action. The section also discusses also social media platforms, such as Facebook and Instagram, and explains why Twitter is likely to remain the dominant source of social network data for
social scientists.

Section 6 concludes with several discussions of Twitter as a data source. The section discusses limits to acquiring data and making inferences given the little information provided in individual tweets. It discusses potential ethical concerns with tweets, especially as it concerns protected groups of people such as children. Finally, it concludes that Twitter exhibits characteristics of a media platform and social network, meaning social scientists can use it to study both media, elite individuals, and normal people.

1.1 Why Twitter?

Six features of Twitter have driven its popularity for academic study. First, it is one of the largest social networks, with 284 million active users in almost every country and over $1 billion of annual revenue (Twitter, 2014). These users include heads of state, companies, non-profit organizations, international non-governmental organizations, celebrities, athletes, journalists, academics, and, primarily, normal people. In the United States, as of November 2016, 24% of adults use Twitter; men and women use it equally, a plurality of users are between 18-29 years of age (a majority, 18-49), a plurality of users have at least a college degree (20% have a high school degree or less), a majority earn more than $50,000 per year, and users are evenly distributed across urban, suburban, and rural areas. 42% of its users use it daily and 24% at least once a week (versus 76% and 15% for Facebook) (Duggan, Page and Manager, 2016). Twitter therefore provides a cross-section of almost any group in which a researcher would find interest.

Second, users produce a lot of data, 500 million messages per day. All these people and messages mean that Twitter mirrors vast segments of the population that would otherwise require large teams of researchers to analyze concurrently. Taken together, these first two characteristics mean that almost any event is recorded on Twitter, and many events are predictable with it. For literature on event prediction using Twitter, see Table 1 at the end of Section 1.2.
Third, it makes these data relatively easy to obtain. Twitter makes its users’ data through two APIs, the streaming and REST API, that are accessible to anyone with a Twitter account. Before the switch to v1.1 of the APIs, Twitter allowed third parties to provide interfaces that allowed individuals with no programming experience to access its APIs. Now, that capability no longer exists, raising the barrier to entry for acquiring data. Nonetheless, because of Twitter’s popularity, there are a large number of software libraries to access Twitter, including in Python and R. One needs some programming knowledge to interact with the APIs, though not as much as just a few years ago. The primary purpose of this Element is to help the reader gain that knowledge.

Fourth, the APIs make it easy to tailor the data one receives to a specific research question. One can receive as much as 1% of all tweets every day (from the streaming API) or filter the tweets received based on keywords, user location, user IDs, or language used. Through the REST API, one can download specific tweets, 3,200 of a user’s most recent tweets, a list of who a user follows or who follows the user, and user profile information. In other words, though Twitter has pioneered many “big data” technologies, one does not necessarily need to possess these skills to access Twitter’s data. One may need to learn new skills to gather the data, but modeling and visualizing those data can be done with old tools of the trade.

Fifth, Twitter is an excellent data source for network and non-network analysis. Since the service explicitly structured as a network - connections between accounts are the fundamental building blocks of the user experience - researchers interested in diffusion processes and emergent behavior find Twitter a natural source. But Twitter, because its 1% stream delivers tweets without information on the tweet author’s social network, is also a compelling source for researchers interested in polling and event prediction (Gayo-Avello, 2013).

Sixth, Twitter has a norm of public conversation that does not exist on Facebook. While Facebook also provides an API, most users choose not to make their information publicly available. To gain access to a user’s information, one needs to design a Facebook app that
the user installs or work with Facebook’s research team. This team maintains veto power over research proposals and publications, and the recent controversy over manipulation of Facebook feeds has caused Facebook to tighten control over its research team (Kramer, Guillory and Hancock, 2014).

1.2 Types of Questions

Twitter data can be used to answer questions that involve three kinds of data: networks, text, and spatial.

Perhaps the most exciting potential of Twitter is as a tool for reconstructing social networks. Networks can be reconstructed from streaming data or data downloaded through the REST API. Because the REST API provides follower and friend information, only it permits the reconstruction of all of a user’s connections. A researcher interested in the complete social network of an individual therefore has to use the REST API.

Computationally analyzing text is a growing field in the social sciences (Grimmer and Stewart, 2013; Lucas et al., 2015), and Twitter provides scholars with large corpora. The barriers to entry for Twitter are substantially lower than for appearing in a news report or in Congress. People who are not likely to appear in those sources are therefore much more likely to appear on Twitter, casting light on vast swathes of previously unobservable behavior. While this outcome is true of the internet more broadly - barriers to entry are similarly low for joining most social networks, posting to web forums, or starting a blog - Twitter is unique in its combination of size and public communication. Elites who are likely to appear in traditional text sources usually have Twitter accounts (every Senator and 430 members of the House have official Twitter accounts, as do most news organization), meaning Twitter records communication behavior from all strata of society. No other data source exists which is simultaneously comprehensive and accessible. For literature that applies natural language processing to Twitter, see Table 1.

It is also possible to map Twitter activity to specific places, allowing scholars to connect
patterns on Twitter to offline events. Location information comes from two sources, accounts choosing to provide their GPS coordinates or self-reporting their location as part of their profile. Tweets with GPS coordinates, which represent 2-3% of all tweets (Leetaru et al., 2013), provide the most precision in estimating location. See Table 1 for examples of mapping Twitter to events. Tweets with GPS coordinates are more prevalent in urban areas and among higher-income users, so the extent to which they are reliable depends on the research question (Malik, Lamba, Nakos and Pfeffer, 2015). See the Supplementary Materials for an analysis of which countries produce the most tweets with GPS coordinates, both in absolute and per capita terms.

**TABLE 1 HERE**

### 1.3 A Note on Programming

Twitter data can be acquired, processed, and analyzed with many programming languages, including R and Python. R is the programming language most familiar to political scientists. An extension of S, a language developed at Bell Laboratories in the 1970s, R was designed by statisticians. While it can perform many general computation functions, its comparative advantage is in statistical analysis. When a new statistical procedure is developed, the first implementation is usually as an R package.

R can ingest Twitter data in 3 ways. streamR is a package that makes it easy to connect to R’s streaming API and write the returned tweets to a .csv or JSON file (Barbera, 2013). twitteR is designed to work with the REST API, though the complexity of the REST API compared to the streaming one means the package is not as robust as streamR (Gentry, 2015). One could avoid these packages completely by using the RCurl package, which facilitates interaction with the HTTP endpoints that web services, including Twitter, use (Lang, 2015). Using RCurl provides the most flexibility but requires more coding than using a package designed to work with Twitter.
<table>
<thead>
<tr>
<th>Topic</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prediction</td>
<td>Box office (Asur and Huberman, 2010)</td>
</tr>
<tr>
<td></td>
<td>Coups (Kallus, 2013)</td>
</tr>
<tr>
<td></td>
<td>Crime (Gerber, 2014)</td>
</tr>
<tr>
<td></td>
<td>Memes (Garcia-Herranz et al., 2014)</td>
</tr>
<tr>
<td></td>
<td>Stock market (Bollen, Mao and Zeng, 2011; Zheludev, Smith and Aste, 2014)</td>
</tr>
<tr>
<td></td>
<td>Unrest (Ramakrishnan et al., 2014; Steinert-Threlkeld, 2017b)</td>
</tr>
<tr>
<td>Disaster Response</td>
<td>Starbird and Palen, 2010; Vieweg, Hughes, Starbird and Palen, 2010;</td>
</tr>
<tr>
<td></td>
<td>Yardi and Boyd, 2010</td>
</tr>
<tr>
<td>Polarization</td>
<td>Barberá, Jost, Nagler, Tucker and Bonneau, 2015; Borge-Holthoefer, Magdy,</td>
</tr>
<tr>
<td></td>
<td>Darwish and Weber, 2015</td>
</tr>
<tr>
<td>Congress</td>
<td>Barberá, Bonneau, Egan, Jost, Nagler and Tucker, 2014; Anastasopoulos,</td>
</tr>
<tr>
<td></td>
<td>Badani, Lee, Ginosar and Williams, 2016</td>
</tr>
<tr>
<td>Demographics</td>
<td>Hale, Gaffney and Graham, 2011; Zamal, Liu and Ruths, 2012a; Mocanu,</td>
</tr>
<tr>
<td></td>
<td>Baronchelli, Perra, Vespignani, Goncalves and Zhang, 2013</td>
</tr>
<tr>
<td>Economics</td>
<td>Acemoglu, Tahoun and Hassan, 2014; Llorente, Garcia-Herranz, Cebrian and</td>
</tr>
<tr>
<td></td>
<td>Moro, 2014</td>
</tr>
<tr>
<td>Geography</td>
<td>Yardi and Boyd, 2010; Kulshrestha, Kooti, Nikravesh and Gummadi, 2012;</td>
</tr>
<tr>
<td></td>
<td>Conover, Davis, Ferrara, McKelvey, Menczer and Flammini, 2013; Frank,</td>
</tr>
<tr>
<td></td>
<td>Mitchell, Dodds and Danforth, 2013</td>
</tr>
<tr>
<td>Sentiment</td>
<td>Dodds, Harris, Kloumann, Bliss and Danforth, 2011; Golder and Macy, 2011a</td>
</tr>
</tbody>
</table>

Python is a general purpose language tracing its lineage to 1989 and is most famous for having easy to read code. Whereas R was created for data analysis and has been extended
to other purposes, Python was created to work with computers and has been extended to
data analysis. Transitioning from writing in R to writing in Python for the first time is much
easier than transitioning from never having written code to writing in R.

The primary Python library for working with Twitter is twython (McGrath, 2015).
tweepy is another Python library to access Twitter, but twython has a larger community and
is more frequently updated.) Unlike any R package, twython can work with Twitter’s REST
or streaming API and has built in exception handling. Python’s pandas library provides data
frames equivalent to R’s as well as reshaping, merging, and aggregation capabilities spread
across multiple R packages (McKinney, 2015); pandas is much faster than base R, though
R’s data.table package is as fast to slightly faster than pandas (Dowle et al., 2015). Python’s
statistical libraries are not as deep as R, though most parametric and non-parametric mod-
els are available through the statsmodels package (Seabold and Perktold, 2014). Libraries
for Bayesian analysis are not as developed, though Stan has a Python interface. Python
has extensive libraries for natural language processing (Bird, 2015) and machine learning
(Mueller, 2015) as well.

Neither R nor Python strictly dominates the other. Python has more developed tools
for scraping web pages, but Hadley Wickham’s rvest package narrows this gap. Python
is generally faster, but new R packages such as data.table erase that difference on some
dimensions. R’s syntax does not resemble other computer languages’, but it is also easy
to read and learn. The one area Python dominates R for data analysis is data storage:
many more database products have Python libraries than R ones, though R has libraries
for working with SQL, SQLLite, and MongoDB (a prominent NoSQL database). That said,
most people are unlikely to need a database for their Twitter work. R dominates Python in
developing aesthetically pleasing graphics, though a Python port of ggplot is being developed
and Python’s matplotlib library produces Matlab style graphics. If one already knows R,
learning Python may be worthwhile, but the costs and benefits require careful consideration
since either language can most likely accomplish one’s programming task and human time is
the most scarce resource on any project. The more likely one is to work with large amounts of data or colleagues from outside the social sciences, the more beneficial Python knowledge becomes. If one knows neither R nor Python, learn both.

2  Acquire Data from Twitter

There are 3 approaches to acquiring data: using Twitter’s API, collaborating with those who have collected them, or purchasing them. This section details each in turn.

2.1  Acquire on Your Own

The first approach, and the one that is most likely to satisfy one’s research question, is to download the data yourself. Advantages of this approach include being able to define search terms, not relying on others for data, and, depending on how much data is involved, cost. Disadvantages include a steeper learning curve than purchasing or working with others, difficulty accessing historic data, and needing to maintain one’s own infrastructure. Section 2.1.1 and 2.1.2 explain the two application programming interfaces (APIs) for acquiring data for free and what kinds of data are available from each. Though Twitter does not charge for using those interfaces, one still needs hardware with which to store and analyze the data.

2.1.1  REST API

The REST API provides access to past tweets, user data, and social structure. Below, each paragraph explains the type of data the REST API provides. The italics at the beginning of the paragraph is the type of data; the paragraph then explains each type in detail.

User’s Tweets Twitter allows anyone to download an account’s previous 3,200 tweets through the GET statuses/user_timeline endpoint. This endpoint accepts 180 requests per 15 minutes and returns up to 200 tweets per request. An account with 3,200 or more tweets will therefore require 16 requests. With each tweet, Twitter returns metadata on the tweet
author, but that metadata reflects when the API request was made, not when the tweet was created. @Greptweet is an interface on top of this endpoint.

Specific Tweets One can download more than 3,200 of a user’s tweets if the identification number of each tweet is known. Since Twitter’s Terms of Service prevent researchers from sharing more than 50,000 original tweets per day but allows one to share an unlimited number of tweet IDs, a researcher who would like to replicate other work or use previous tweets in original research is therefore reliant on the goodwill of the original acquirer of the tweets and having the programmatic ability to download the tweets. (See Freelon (2012) for an example of freely shared tweet IDs.) Fortunately, Twitter’s rate limits are generous for downloading tweets, and the code to do so is simple. One can download 18,000 tweets every 15 minutes (100 for each of 180 calls), equivalent to 1,728,000 per day. Downloading user information is subject to the same limits.

If one would like to download more than 1.7 million tweets per day, there are two options. The easiest approach - split the list of IDs into small chunks and submit those chunks at the same time - will not work because Twitter only allows 1 connection per IP address. The two options for downloading tweets at a higher rate are the two options to get around the IP limit. The first approach, which violates the spirit of Twitter’s limit, is to route the requests through proxy servers. The second option is to use multiple computers. It is not difficult to launch multiple virtual instances using a hosting solution such as Amazon Web Services; the main drawback with a hosted solution is cost. Cost is substantially lessened if you use Amazon’s t2.small instances and push the Twitter data to a local machine. Each t2.small instance costs $137-$180 annually, though you will also pay to transfer data. Otherwise, friends’ computers or old machines are perfect for this sort of task; since downloading data is not computationally expensive, old computers are perfect for any Twitter task where rate limits force the task to take a long time.

Search Tweets It is possible to query Twitter’s GET search/tweets for old tweets matching certain parameters. This method is to be used cautiously, however, as Twitter returns only
some tweets from the previous 6-9 days and is not clear on how it chooses which to return. Only 100 tweets per request are returned, up to 180 requests per 15 minutes. To avoid receiving the same 100 results per request, pass the lowest tweet ID of the returned tweets to the max_id parameter in the subsequent call, and repeat this process as much as necessary. In addition to search terms, Twitter allows for filters based on language and latitude and longitude pairs; one can also specify for Twitter to return the most recent or most popular matching tweets, or a mix. One study has found that results from GET search/tweets do not match those from the random sample (González-Bailón et al., 2012).

User’s followers Reconstructing network connections is slightly more difficult. Two endpoints, GET followers/list and GET followers/ids, provide information about followers. The former provides fully hydrated user objects for each follower, up to 15 followers per 15 minutes. The latter provides only the identification number of followers, but it does so for 75,000 followers per 15 minutes; those numbers can then be fed to GET users/lookup, from which up to 18,000 completely hydrated user objects are returned every 15 minutes. GET followers/list therefore saves one step but is slower than using GET followers/ids with GET users/lookup.

User’s friends The same logic holds for retrieving who a user follows (that user’s friends, in Twitter parlance). One connects to GET friends/ids instead of GET followers/ids, but those friend identification numbers are fed to GET users/lookup.

Twitter returns the follower and friend list in reverse chronological order but does not reveal when either connection is formed. Section 3.2.4 explains how to infer connection dates using the REST API.
2.1.2 Streaming API

This section details how to collect data in realtime via Twitter’s streaming API.\textsuperscript{1} There are two levels of access available, the 1\% vs. 10\% stream. Twitter removed free access to the 10\% stream - variously called the garden hose or fire hose - in early 2011, and one now has to apply and pay for access. (Twitter does not disclose the price of connecting to the garden hose.) The 1\% stream remains free and probably will be forever, as it is the connection developers use to build Twitter-related products. For documentation on the API, see the Twitter developer guidelines. The italics that start each paragraph indicate variations of the 1\% stream. Because the streaming API returns all tweets matching filtering parameters so long as the total number of returned tweets is less than 1\% of all tweets, these variations are different sets than a subset of tweets from the pure random sample matching those parameters.

Random sample By default, Twitter returns a random 1\% sample of tweets as they are written. This sample comes out to about 5 million tweets and 12 gigabytes of raw data per day. Because each returned tweet object consists primarily of metadata, preprocessing or post processing the tweets can reduce storage requirements without reducing the amount of useful data. Twitter does not disclose how it chooses the sample, leading to concern about the representativeness of the stream versus the complete Twitterverse. Comparisons of the stream to the complete Twitterverse find the stream does not differ in a meaningful manner (Morstatter et al., 2013; Valkanas et al., 2014).

One can also request Twitter filter the results from the streaming endpoint. If a filter is given, Twitter returns all tweets matching the request \textit{up to 1\% of all tweets}. For example, if one asks for every tweet with the keyword “LeBron” and tweets with that word constitute .1\% of all tweets, Twitter will return every single tweet containing that word. This design is extremely advantageous for researchers, as it means the streaming sample can often become

\textsuperscript{1}The streaming API technically has 3 endpoints: GET statuses/sample, GET user, and GET site. Academics will only need to work with GET statuses/sample, so that is the connection assumed for the rest of this Element.
a streaming population. With filters, the streaming API can provide a researcher with every
tweet of interest, though the researcher will have to know ahead of time what filters are of
interest. To continue the keyword example, if the researcher connects to the stream without
specifying “LeBron” only 1% of all tweets containing that word will be returned, 2 order of
magnitude fewer tweets than requesting tweets specifically with “LeBron”.

*Geographic sample* The streaming API can also return tweets from within a box defined
by two coordinate pairs. The bounding boxes are not used in conjunction with other filters.
For example, asking for tweets from San Francisco and tweets in Spanish will return all tweets
from San Francisco (regardless of language) and all tweets in Spanish. Since 2-3% of tweets
contain GPS coordinates (Leetaru et al., 2013), passing the coordinate pairs [-180,90,180,90]
- a box around the world - will return 33% to 50% of all tweet with GPS coordinates. Twitter
accepts up to 25 bounding boxes per connection. The streaming API does not use a user’s
self-reported location.

*Specific Keywords* Twitter will return tweets containing a user-supplied string, and mul-
tiple strings can be passed. This functionality can be used to search for specific hashtags,
individual words, links (Twitter will search the expanded URL of a shortened link), retweets,
or mentions of a user. 400 pieces of text can be passed per connection. Note that non-space
separated languages, like Korean, Japanese, and Chinese, are not supported.

*Language sample* When connecting to the streaming API, one can request only tweets in
a certain language. Twitter will then return all tweets in that language up to the 1% ceiling.
Note that language cannot be the only parameter passed. To download tweets in a specific
language, it is therefore best to pass generic keywords in that language, e.g. “this”, “the”,
“is”, and so on if one wants a sample of tweets in English. The request will then return a
random sample of the tweets in English that contain one of those words. Multiple languages
can be requested simultaneously. Because Twitter does not filter for non-space separated
languages, asking for tweets in those languages requires use of other parameters.

*Specific People* One can submit to the streaming API specific user identification numbers
and receive all tweets the users create, all tweets the users retweet, replies to tweets of the
users, and retweets of the users' tweets. 5,000 people can be followed per connection. This
feature is especially useful when the accounts to be studied are known. The best way to
identify accounts is through lists, curated collections of accounts other users have created.
For example, Twitter maintains the “US Senate” list, a list of the Twitter accounts for each
Senator (every Senator has a Twitter account). A researcher could ask the REST API for
the user identification number of each member of this list and then pass those numbers to
the streaming API. Every tweet from every Senator would henceforth be downloaded by the
user, assuming the 100 Senators never account for more than 1% of all tweets.

Except for the GPS bounding boxes, Twitter’s documentation is not clear on how these
parameters interact with each other. More likely than not, they are additive, e.g. following
a specific user and asking for tweets in Spanish will probably return tweets in Spanish or
tweets from that user.

2.1.3 Access Constraints and Replication

While Twitter’s streaming and REST APIs are powerful, they have 7 quirks to be aware of
when writing code. These quirks render some kinds of analyses, especially those relying on
the REST API, more difficult than they otherwise would be. They also raise the costs of
replication, though full replication remains possible.

First, an overarching restriction is that Twitter imposes limits on how quickly one can
retrieve data from the REST API. (The only restriction of the streaming API is that not
every tweet is returned if the number of matching tweets exceeds 1% of the total volume
of tweets.) Requests are counted in 15 minute windows, and most API endpoints allow 15
requests per window. Each request, however, may return multiple matches, and the number
of matches returned is not constant across endpoints. For example, Twitter returns 5,000
followers per request, so 75,000 followers can be downloaded in one window; when asking
Twitter for metadata about an account, 100 accounts per request and 180 requests per 15
minutes are allowed, allowing for metadata on 72,000 accounts every 15 minutes. Download time is a step function. The amount of data returned in each request is small, so Twitter can return the results from 15 requests in a matter of seconds. A user with 75,000 followers therefore requires a few seconds to download the list of followers, but one with 75,001 requires just over 15 minutes.

If one’s code needs to query an endpoint more than the rate limit allows, the code must restrict itself; otherwise, Twitter will sever the connection and reserves the right to ban the account from querying the APIs again. The rate limits make it very difficult to reconstruct complete social networks, an issue returned to in Section 3.2.4. Details on rate limits for specific endpoints is available at Twitter’s developer website.

Second, Twitter only allows one connection per IP address to any part of its API. For example, downloading the 1% stream and a separate stream returning only tweets from the United States would require two separate IP addresses. Similarly if one wants to connect to the stream while parsing accounts’ followers. Without this restriction, the rate limits would be meaningless. Nonetheless, the ability to create virtual machines on demand, using a product like Amazon Web Services, makes the IP address restriction less onerous than it otherwise would be.

Third, when asking for an account’s previous tweets, the REST API returns only the 3,200 most recent tweets from an account. This restriction means that one will not obtain, for free, all the tweets from accounts which tweet often, such as celebrities, politicians, or media accounts. Because there is a positive relationship between tweet frequency and number of followers (Gonzalez-Bailon, Borge-Holthoefer and Moreno, 2013), the limit means that a sample of users for whom all the tweets are available is a sample of less popular accounts. Moreover, the metadata of each tweet is not reflective of the tweet at the time it was authored. For example, a tweet from the streaming API will show how many followers the author has when the tweet is created, but that same tweet downloaded from the REST API will show the number of followers the account has at the time the old tweet was downloaded. This
wrinkle means that tweets from the REST API are not equivalent to tweets from the stream.

Fourth, while the REST API allows one to search historic tweets, the results are only from the previous 7-9 days and not exhaustive of those days. Twitter does not explain how it decides which tweets to return, so it should not be relied upon to reconstruct histories. Twitter only returns 100 results per request, up to 180 requests per 15 minutes. A broad search with thousands of results may therefore take awhile to download and will not provide the population of tweets matching a search query. Searching directly at www.twitter.com returns all historic matches, but one cannot download those matches.

Fifth, Twitter only allows one to share 50,000 public tweets and/or accounts’ metadata per day, and the sharing cannot be automated. For example, if a researcher uses more than 50,000 tweets for a paper and needs to share them, he or she cannot make them freely available. A system would need to be constructed to verify that an interested party is not downloading the data more than once per day, and the data cannot be pushed to an interested party. Twitter does allow, however, the unlimited distribution of the numeric identification number of each tweet or user account. An interested party can then take these numbers to the REST API and download the full tweet and account information.

Sixth, the streaming API occasionally disconnects. These disconnections are rare and random but can imperil one’s research design if not caught quickly. At least three solutions are available. If one’s connection is designed to last indefinitely, that connection’s code should generate an e-mail, or similar notification, whenever the connection is interrupted. Alternatively, one can intentionally disconnect from the streaming API and immediately reconnect at preset intervals, such as every one to twenty-four hours. Any stoppage of the stream would therefore impact only that interval’s data collection. (This strategy is what I employ in my data collection. I collect tweets in one hour intervals, using Unix systems’ cronjob scheduling. My server launches a new script every hour, and that script

---

2Technically, the number of requests depends on whether one is authenticated as a user or application. Since whether or not one form of authentication returns more results depends on the type of request and most academics are not trying to build an application, the rate limits presented are for the user authentication.
only downloads tweets during those 60 minutes. Any disconnect in the previous hour will therefore cause me to lose no more than 60 minutes’ data.) Third, you could implement a monitoring script that will restart the script whenever a disconnect notification is generated. This approach results in the least amount of data loss; the streaming API is very reliable, however, so the researcher should weigh the extra time implementing this check will require versus the hours of data it may save.

Seventh, when requesting specific users, best practice calls for using an identification number. When a user changes his or her username, Twitter does not update the user identification number corresponding to the original screen name. Asking either API for user information based on the screen name may therefore suffer from decay as users change their name, while asking by user identification number will not.

2.2 Collaborate

On April 14, 2010, Twitter and the Library of Congress announced the Twitter Research Access project, a collaboration to make every tweet ever published available to researchers (Stone, 2010). Scheduled for completion in 2013, the project still has not resulted in an available archive. Updates from the Library have been intermittent, though it is clear it has at least all tweets from 2006, when Twitter started, through the end of 2010. Disconcertingly, a report from the Library in 2013 suggested that the hardware necessary to enable fast searches of the archive are “cost-prohibitive and impractical for a public institution” (Update on the Twitter Archive At the Library of Congress, 2013). The most detailed report on the project and its current status is to be found in Zimmer (2015).

@Greptweet hosts greptweet.com, a website that returns a user’s 3,200 most recent tweets. This website is the most user friendly method to get a user’s tweet history, but it suffers two main disadvantages. First, it only returns the tweet identification number, tweet timestamp, and tweet text; while one can retrieve the full tweet based on this information (see Section 2.1.1), it would be preferable to have the data without programming. Second, one can only
get tweets for one user at a time, which is a much slower process than submitting requests through the API. greptweet.com is therefore best for retrieving data for a small group of people who do not tweet frequently.

TCAT allows an interested party to connect to Twitter’s streaming API, follow users or keywords, and create network graphs through Gephi, all without any programming expertise (Ward, 2014). TCAT is best for identifying accounts which are influential in talking about a particular product. The service is no longer open to the public, though inquiries are accepted on an ad hoc basis (Groshek, 2015).

CrisisNet is part of Ushahidi, an open source digital platform that supports data-gathering from the internet and mobile phones, with a focus on the developing world and public sector implementations. CrisisNet ingests data from Ushahidi, Facebook, Twitter, Instagram, and ReliefWeb, processes that content into a standard form, and makes each post from each source available through an application programming interface. It is not a complete archive of content from the data sources it monitors; instead, it pays attention to specific accounts that CrisisNet users identify. Once an account is identified, its data from that point forward is part of CrisisNet. The platform is still a small operation and has the steepest learning curve of these tools.

Finally, many academics have already collected, or collect continuously, their own Twitter data, though Twitter’s terms of service prevent the mass sharing of tweets. Collaborating with someone who already has Twitter data on a topic or timeframe of interest is often the quickest, most inexpensive method of acquiring tweets.

### 2.3 Purchase

If one is interested in tweets from the past, the most thorough approach is to purchase them from a vendor. (Section 2.1.1 describes how to download some old tweets for free.) Companies which provide access to old tweets pay a large, undisclosed licensing fee to Twitter, and their main customers are marketing and public relations firms. Since many companies
provide this service, this section focuses on four of the most prominent.

The main vendor is Gnip; it was founded in 2008 and started licensing Twitter’s data in 2010, and Twitter purchased the company outright in 2014. While Gnip’s target market is businesses, anyone can purchase their old Twitter data using their Historical PowerTrack application programming interface (API). Gnip claims that prices start at $500, but a project will more likely spend upwards of $5,000 purchasing data. The price is a function of the number of tweets that will be returned and the timespan of the request, though the final price requires consultation with a sales representative. Gnip provides its own metadata as well, including expanded links, a Klout score, language detection, and enhanced geo-information. Though Gnip provides a programming interface, a one-time purchase is best handled through contacting a salesperson directly. Gnip has offered Data Grants, free downloads of tweets to winning applicants; its first, and so far only, competition saw 1,300 entrants compete for 6 grants.

DataSift is a third major reseller of archived tweets. They provide the same services as Gnip and Topsy, and, like those services, provide a programming interface that uses their own syntax to filter historic data. (DataSift, Gnip, and Topsy also ingest other datasources such as Wikipedia, Reddit, YouTube, and WordPress.) DataSift’s historic data starts on January 1st, 2010, whereas Gnip and Topsy have every tweet since 2006. DataSift charges $1 per 5 hours of computation time, plus $.10 per 1,000 tweets a query returns. To estimate the cost of a dataset, the researcher has to submit a query to DataSift’s API. DataSift will return the estimated number of matching items, which, when divided by 10, will give the cost in dollars for the tweets themselves; the final cost will also include computation time. Two features distinguish DataSift from its competition. First, it accesses more services’ data than other services. Second, DataSift has also published a Python library to access its API. An annual subscription costs $200,000 - $300,000.

The most cost effective method to acquire historic tweets is through Texifter’s Sifter tool. Texifter is a 3rd party vendor that interfaces with Gnip. The main advantage of Sifter is
that it provides an online interface to Gnip’s API so that a user can test a variety of search parameters. Sifter will then estimate the number of tweets the download would contain and provide a cost estimate. Sifter also appears to offer a lower price point than Gnip, though whether that is true or not is unclear since Gnip’s pricing structure is opaque. Sifter then transfers the data to Texifter’s DiscoverText tool, an online interface through which a coding team can search, filter, code, or analyze the returned tweets. The tweets can then be downloaded, though Twitter’s terms of service require that Sifter only allow 50,000 tweets per day to be downloaded, regardless of how many match a user’s query.

3 Process Data

3.1 Getting Started

This section takes you through creating a Twitter account and creating an application with that account. The application is what will connect to Twitter’s APIs, and an account can have multiple applications.

Section 3.1.1 explains how to create an account, and Section 3.1.2 shows how to use that account to obtain the credentials you will need to access Twitter’s API. Section 3.2 provides Python and R scripts to download a user’s tweets, download specific tweets by their identification number, search for tweets, download an account’s followers, and download an account’s friends (the accounts the account followers). Section 3.3 provides Python and R scripts to download a 1% random sample of the Twitter stream as well as request tweets from the stream matching language, geographic, account, or keyword parameters. The sections show the key line or lines that download the data and provides links to the full script. Each script is designed to run manually, so there is redundancy in the first lines of each.

3.1.1 Create Account

1. Load Twitter (www.twitter.com) in your internet browser.
2. On the right side of the browser window, you will see a box that says “New to Twitter? Sign Up”. Fill that box with your name, contact e-mail address, and a password.

3. After entering your name, contact e-mail address, and a password, click the yellow button that says “Sign Up for Twitter”.

4. The website will load a new screen. Enter your name, contact e-mail address, and password again. Add your desired username, which is what your primary name will be on Twitter. Uncheck the boxes about staying signed in and tailoring Twitter if you desire.
5. Click the yellow “Create my account” button after completing the previous step.

6. In the screen that loads after clicking the “Create my account” button, follow steps 1-5. These steps provide Twitter with more information about yourself. Since this account will be for research purposes, it is not important to answer these. Note that there is a small phrase, "Skip this step", at the bottom right of your screen; clicking those words will allow you to progress to the next page.

7. After completing those steps, you will arrive on your Twitter homepage. You will see a banner notifying you about a confirmation e-mail and giving you options to resend the e-mail, change your e-mail address, or get help (the “learn more” text).
8. Check your e-mail using the e-mail address you gave Twitter. You will have an e-mail with a blue “Confirm now” button. Click that button. If you do not see this e-mail after a few minutes, check your Spam folder.

9. Clicking the “Confirm now” button will take you to the Twitter homepage for your account. It is worth following some accounts and sending a couple of tweets immediately so Twitter does not delete your account for inactivity.
3.1.2 Register as Application

Before you can request data from Twitter, you will have to prove to Twitter that you actually have an account. This proving is accomplished through the OAuth authentication standard. This section will show you how to authenticate yourself with Twitter.

1. Use your internet browser to load www.dev.twitter.com. If you are not still logged in from Step 1, you will see a screen similar to the following screenshot. Twitter changes their background image often, so do not worry if your webpage does not look exactly like this:

2. Move your mouse to “Developers.” A menu will drop down. Click on “Documentation”.

28
3. A new page will load after clicking “Documentation”. On the left navigation bar of that new page, click “Manage My Apps”.

4. Twitter will tell you that you do not have any applications yet. Click “Create New App”.

5. The next screen is where you create your application, which is what you will use to access Twitter’s API. Fill in the fields with whatever words you want, though it is best that the name and description correlate with the purpose of the application. The website field can be any website, unrelated to you or not. Once the fields are completed, check the “Yes, I agree” box for the Developer Rules of the Road and click the “Create your Twitter application” button.

---

You should also read the rules of the road. They are short and explain the limits to sharing tweets.
6. After creating your application, you will find yourself at the main page for your application and see four navigation tabs - “Details”, “Settings”, “Keys and Access Tokens”, and “Permissions”. Click on “Keys and Access Tokens”.

7. After clicking on “Keys and Access Tokens”, your screen should look like the one below, with your application’s name in large, bold font. Note that the application name has changed from “ExampleApplicationNeophytes” to “ExampleApplication” because the latter was taken.
8. Copy and paste the “Consumer Key (API Key)” and “Consumer Secret (API Secret)” fields somewhere you can easily retrieve them, like in a text document. These will be necessary soon. The method of connecting to the streaming API through R does not require the next steps; if you are using R, skip to “Use R to Verify Your Identity”. Libraries in other languages will accept the two items created in the next steps or let you replicate the steps R requires.

9. At the bottom of the screen, click the “Create my access token” button.

10. Once you click the “Create my access token” button, the screen will refresh, with more information displayed under the “Your Access Token” section. If there is no
11. Copy and paste the “Access Token” and “Access Token Secret” fields somewhere you can easily retrieve them, like in a text document. You will also need these shortly.
3.2 Using the REST API

R’s twitteR package is designed to work with the REST API and is used in the following examples. Its documentation is not user friendly, but there are many examples online of how to use it to accomplish common Twitter tasks. An alternative option is to access the API directly through PycURL or RCurl; those libraries use Python and R to work with curl and libcurl, lower level tools to transfer files via the internet. Working directly with the API entails submitting requests as long URLs and provides the most flexibility to the researcher. Since that approach is has a steeper learning curve than working with a library designed for Twitter, it will not be presented here.

3.2.1 User’s 3200 Tweets

The full R script to download a user’s tweets is here (NB: Link to RESTAPISection_twitteR_userTweets_v1.R). The key line is this:

```r
tweets <- userTimeline(user, n = 3200, includeRts = TRUE)  # n = 3200 ensures all tweets will be downloaded; Twitter does not return more than 3200 tweets.
```

tweets is a list, and the twListToDF() function converts that list to a data frame. The script also shows how to save the tweet identification numbers to a text file, a necessary step for others to replicate your work.

The full Python script to download a user’s tweets is here (NB: Link to RESTAPISection_twython_userTweets_v1.py). The script defines a function to retrieve a user’s tweets using either the screen name or identification number, and it can be easily extended to handle multiple users at once. It also contains a function to save the tweets to a file. The key lines are:

```python
temp = connection.get_user_timeline(screen_name=screen_name, count=200, max_id =maxID - 1)
```
tweets = getUserTweets(screen_name='ZacharyST')
saveTweets(tweets, filename='<insert your name here>')

### 3.2.2 Tweets by ID

One can download specific tweets if the tweet identification number is known. This feature is useful when if one wants to replicate other Twitter studies: Twitter does not allow one to share more than 50,000 raw tweets per day, but one can share an unlimited number of tweet identification numbers. For tweet identification numbers, Twitter returns 6,000 tweets per 15 minutes, or 576,000 per day. For an example of a researcher sharing tweets, see Freelon (2012).

The full R script to download specific tweets is here (NB: Link to RESTAPISection_twitteR_specificTweets_v1.R). The key lines are:

```r
# Load one tweet
tweet <- showStatus(tweet_IDs$ID[1])

# Multiple tweets require more work
chunkedTweets <- split(tweet_IDs$ID, ceiling(seq_along(tweet_IDs$ID)/submissionsPerRequest)) # ceiling(seq_along(tweet_IDs$ID)/submissionsPerRequest) generates a list of repeating numbers for split() to split. split then creates a list where each entry is as long as submissionsPerRequest (100 in this example).

tweets <- NULL # Empty object that tweets will feed into

# Download tweets
for(i in 1:length(chunkedTweets)){
    print(paste('On cycle ', i, ' of ', tweets, sep='')) # Status tracker
    # Sys.sleep(delay) # How many seconds to pause so that do not trip rate limit. Commented out in this loop because downloading 3,200 tweets will never exceed the rate limit. (60 requests * 100
```
tweets per request) > 3200 tweets

tests <- twListToDF(lookup_statuses(ids = chunkedTweets[[i]]))
tweets <- rbind(tweets, temp)
}

The full Python script to download specific tweets is here (NB: Link to RESTAPI-
Section_twython_specificTweets_v1.py). The key lines are:

# Download tweets
for chunk in chunkedTweets:
    print(’On new chunk’)
    temp = connection.lookup_status(id=chunk) # Notice that this is
    lookup_status and not show_status
    tweets.extend([item for item in temp])

3.2.3 Search tweets

Twitter’s REST API also allows one to search for recent tweets by calling the GET search/tweets
endpoint. The parameters one can pass are quite similar to the parameters the streaming
API accepts, but the search function only returns tweets from the previous six to nine days.
(Twitter’s documentation variously states the previous seven days or the previous six to
nine days.) If one is interested in tweets related to an unexpected event, it is easier to con-
nect to the streaming API and pass the same parameters one would to GET search/tweets.
GET search/tweets is therefore best used to observe the world in the period just before an
unexpected event happens, not to watch the world in real time.

The search functionality treats geolocation different than the streaming API. In the
streaming API, one passes a bounding box, and Twitter returns tweets with GPS coordinates
from within that box. When searching for tweets, one supplies a point and a radius around
that point (in miles or kilometers), and Twitter returns tweets from within that circle. In
addition, the streaming API only returns tweets with GPS coordinates, whereas the search
function will return matches based on a user’s self-reported location. For example, a user
who says they are from New York City, New York and tweets without GPS coordinates will have their tweet returned via the REST API search but not through the streaming API.

While this approach would appear to advantage the search endpoint, that endpoint’s location functionality is circumspect. The example in this section’s script (NB: Link to RESTAPISection_twitteR_searchTweets_v1.R) asks for tweets within 5 miles of New York City that contain the acronym “NBA”. None of the tweets contain GPS coordinates, and the self-reported location of many of the users is unmistakably not within 5 miles of the city. For example, one user is from “Philly”, another from “ALABAMA”, and another from “Watching a Game Somewhere.”. The original tweets can be found at this link (NB: Link to nba_geolocated_tweets.txt).

The full R script to search for tweets matching specific criteria is here (NB: Link to RESTAPISection_twitteR_searchTweets_v1.R). The key lines are:

```r
# How many tweets to return per query
size <- 1000  # Modify as needed

# One word
nba_tweets <- searchTwitter('nba', n = size)  # Also returns hashtags, and Twitter is not case sensitive.
nba_tweets <- twListToDF(nba_tweets)  # Could shorten these two lines to nba_tweets <- twListToDF(searchTwitter('nba', n = size))
```

The full Python script that does the same is here (NB: Link to RESTAPISec-

```
twython_searchTweets_v1.py). The key lines are:
```

```python
# Connect
connection = twy.Twython(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET)

# How many tweets to return per query
size = 100  # Modify as needed, maximum is 100

# One word
```
Both scripts include examples for searching for multiple terms, excluding a tweet if it contains a term, searching for user mentions, returning tweets with images, returning tweets with links, filtering by language, and filtering by geolocation. They also contain a simple example of using the search functionality to build a dataset in real time.

### 3.2.4 Followers

The full R script to download a user’s followers is here (NB: Link to RESTAPISection_twitteR_followers_v1.R). The key lines are:

```r
account <- getUser('ZacharyST')
total_followers <- account$followersCount
account_followers <- twListToDF(account$getFollowers(n = total_followers) #
Create data frame from list of followers of account
```

The R script provides code for getting the followers of one account but not of multiple accounts. The simplest approach to automating the followers download for multiple accounts is to generate a function that applies lines 30 - 77 of the script to each account in the list of accounts whose followers you want. A faster approach, however, is to download the follower identification numbers of all followers of all accounts and deduplicate before downloading the complete user profile. That exercise is left to the reader.

The full Python script to download a user’s followers is here (NB: Link to RESTAPISection_twython_followers_v1.py). The key lines are:

```python
### Get followers for a very popular account
account = connection.lookup_user(screen_name=['BarackObama'])
account_followers = account[0]['followers_count']
get_followers(user='BarackObama', cursor=-1, total_followers=account_followers)
```

37
As with the R script, this script can easily be extended to retrieve followers for multiple users. That exercise is left to the reader.

The scripts show how to heed rate limits when an account has many followers. Downloading followers from an account with more than 5,000 will exceed the rate limits unless the script accounts for those limits.

### 3.2.5 Following

The full R script to download a user’s friends is here (NB: Link to RESTAPISection_twitteR_following_v1.R). The key lines are:

```r
account <- getUser('ZacharyST')
total_friends <- account$friendsCount
account_friends <- twListToDF(account$getFriends(n = total_friends)) # Create data frame from list of friends of account
```

The full Python script to download a user’s friends is here (NB: Link to RESTAPISection_twython_following_v1.py). The key lines are:

```python
## Get friends for a very popular account
```
account = connection.lookup_user(screen_name=['BarackObama'])

account_friends = account[0]['friends_count']

get_friends(user='BarackObama', cursor=-1, total_friends=account_friends)

############ Hydrate friends list. Below function just get list of Twitter IDs.
ids = openIDs(user='BarackObama')

total = len(ids)
i = 0

while i < total:
    print(("On_friend_%d") % i)
    j = i + 100
    hydrateFriends(user='BarackObama', IDs=ids, start=i, end=j)
    i += 100
    pct_done = (j / total) * 100
    print(("Finished_%f percent") % pct_done)

3.3 Using the Streaming API

This section provides scripts for how to connect to Twitter’s streaming API and pass various parameters to it. Note that each R script uses your consumer key and consumer secret strings to generate an OAuth access token, but you only have to do that once. For example, if at t=0 you generate an access token, at all future points you can load that access token instead of generating a new one. The Python scripts do not create an access token, requiring you to submit your key and secret strings each time.

In each R script, the argument ‘timeout = 30’ is used, which tells streamR to download tweets for 30 seconds. This parameter should be modified for your needs; 0 means maintain
the connection indefinitely, and the maximum value is 10800 (3 hours). In my connections, I maintain the connection for one hour (‘timeout = 3600’) and use a service called cron to start the script every hour. (cron is a Unix utility already installed on OS X and most distribution of Linux. See this page for information on a cron equivalent in Windows. Here is a tutorial on cron jobs, and here are the Google results for “cron tutorial”.) This way, any break of the stream will cause one to miss no more than 60 minutes of data. This approach requires file names to change dynamically, and this script shows how to create a function to do that (NB: Link to ScriptsAndExamples/StreamAPISection_streamR_function_v1.R).

My scripts include the time the connection was created in the file name. This naming convention makes it very easy to later find tweets within a specific timeframe. Finally, if you use cron to run the script, you will need to add a “shebang”, a first line which tells the operating system which program to use to execute the following lines.

The longer you maintain a connection, the more data you will miss if the connection is interrupted; for example, making a connection every three hours will lose two hours of data if the connection is severed after one hour. On the other hand, making a connection last for only one minute minimizes the effect of connection interruptions but increases the workload on your computer. If you write one file per connection, you will have 60 times more files than writing every hour. It is not necessarily bad to have so many files, but it can slow down later retrieval of data. One should absolutely not maintain an indefinite connection because the resulting file will be too large to load into RAM, significantly slowing down, and possibly crashing, your computer.

In each Python script, timeout functionality is handled by asking the script to compare the current time to the time when the script started. If that time is greater than a user-defined limit, the script stops. As with R, I have written the script (NB: Link to StreamAPISection_twython_function_v1.py) to collect tweets for 60 minutes, under the assumption that it is launched as a cron job every hour.

The below code extract shows the base code the Python scripts use. In the following
subsections, I then show only the modified `streamConnect` function.

```python
# Import libraries
import twython as twy
import json
import datetime as dt

# Key, secret, token, token_secret for one of my developer accounts.
# Update with your own strings as necessary
APP_KEY = 'yourConsumerKey'
APP_SECRET = 'yourConsumerSecret'
OAUTH_TOKEN = 'yourAccessToken'
OAUTH_TOKEN_SECRET = 'yourAccessTokenSecret'

# Make class
class MyStreamer(twy.TwythonStreamer):
    fileDirectory = '/path/to/directory/to/save/to/'  # Any result from
    # this class will save to this directory

    stop_time = dt.datetime.now() + dt.timedelta(minutes=60)  # Connect to
    # Twitter for 60 minutes. Comment out if do not want it timed.

    def on_success(self, data):
        if dt.datetime.now() > self.stop_time:  # Once minutes=60 have
            # passed, stop. Comment out these 2 lines if do not want
            # timed connection.
            raise Exception('Time_expired')

        fileName = self.fileDirectory + 'Tweets_' + dt.datetime.now().
        strftime("\%Y\%m\%d\%H") + '.txt'  # File name includes
        # date out to hour.
        open(fileName, 'a').write(json.dumps(data) + '
')  # Append
        # tweet to the file

        # NB: Because the file name includes the hour, a new file is
```
3.3.1 Random sample

The full R script to download the 1% sample is provided at this link (NB: Link to StreamAPISection_streamR_randomSample_v1.R). The key line is this:

```r
sampleStream(file.name = 'raw_tweets.txt', oauth = my_oauth, timeout = 30) #
This saves tweets in their raw form as /path/to/your/working/directory/raw
_tweets.txt. Closes after 30 seconds.
```

The full Python script to download the 1% sample is provided at this link (NB: Link to StreamAPISection_twython_randomSample_v1.py). It is the same as the extract provided at the beginning of Section 3.3.
3.3.2 Language sample

The full R script to download a language sample is provided at this link (NB: Link to StreamAPISection_streamR_languageSample_v1.R). The key line is this:

```r
filterStream(file.name='raw_tweets_language.txt', timeout = 30, track = 'a,an, the,and,but,is,this,that', oauth = my_oauth, language = 'en') # Use filler words to capture a large amount of tweets.
```

The full Python script to download a language sample is provided at this link (NB: Link to StreamAPISection_twython_languageSample_v1.py). The key lines are:

```python
# Make function
def streamConnect(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET):
    stream = MyStreamer(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
    stream.statuses.sample(track=['a,an, the,and,but,is,this,that,on,in,up, to'], language=['en'])
```

Note the use of specific keywords in the 'track' argument. Twitter will not return a purely random sample of tweets in a language. Instead, it will use language as a secondary filter to tweets that match other parameters. It is therefore a good idea to use a large amount of generic words. The downloaded tweets are thus a random sample of the tweets containing at least one tracked keyword, and a sufficiently long list of keywords should approximate a random sample of the population of tweets in that language.

3.3.3 Geographic sample

The full R script to download a sample of tweets with GPS coordinates is provided at this link (NB: Link to StreamAPISection_streamR_geographicSample_v1.R). The key line is this:

```r
filterStream(file.name = 'tweets_GPS.txt', timeout = 30, oauth = my_oauth,
             location = c(-180,-90,180,90))
```
The full Python script to download a sample of tweets with GPS coordinates is provided at this link (NB: Link to StreamAPISection_twython_geographicSample_v1.py). The key lines are:

```python
# Make function
def streamConnect(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET):
    stream = MyStreamer(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
    stream.statuses.filter(locations=[−180, −90, 180, 90])
```

Either script downloads tweets with a GPS coordinate from anywhere in the world. Because 2-3% of tweets have GPS coordinates, asking for any tweet with a GPS coordinate will generate a sample containing $\frac{1}{3} - \frac{1}{2}$ of all tweets with GPS coordinates. Narrowing the bounding box is therefore likely to generate a dataset that is the population of tweets from a specific locale. The R script gives examples for downloading any tweet with GPS coordinates, any tweet outside of the United States, and tweets from multiple locations. The Python script shows only a global bounding box, but the coordinates from the R script for the other bounding boxes also work in the Python script.

### 3.3.4 Specific text

The full R script to stream tweets containing certain keywords is provided at this link (NB: Link to StreamAPISection_streamR_keywordSample_v1.R). The key line is this:

```r
filterStream(file.name='tweets_keywords.txt', timeout = 30, track='LeBron
James,Steph_Curry,NBA,basketball,Warriors,GSW,Cavaliers,espn_com', oauth = my_oauth)
```

The full Python script to stream tweets containing certain keywords is provided at this link (NB: Link to StreamAPISection_twython_keywordSample_v1.py). The key lines are:
def streamConnect(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET):
    stream = MyStreamer(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
    stream.statuses.filter(track=['LeBron James, Steph Curry, NBA, basketball, Warriors, GSW, Cavaliers, espn.com'])

Each script downloads tweets about basketball, with an emphasis on the 2016 NBA Finals. Twitter treats multiple word phrases as having a Boolean AND, so a phrase like “LeBron James” will return tweets such as “LeBron James is ready for the game” and “My brother James likes LeBron”; results are not case sensitive, so “lebron james is ready for the game” matches as well. “espn com” is the recommended way to download any tweet from the espn.com domain. Finally, note there are no spaces after the commas; Twitter will treat those as characters to match, so “Steph Curry is ready for the game” would not match if the term passed is “[...], Steph Curry, [...]”.

3.3.5 Specific people

The full R script to download tweets in real time from specific accounts is provided at this link (NB: Link to StreamAPISection_streamR_keywordSample_v1.R). The key line is this:

```r
filterStream(file.name='tweets_accounts.txt', timeout = 30, oauth = my_oauth, follow = '813286,1536791610') # @BarackObama, @POTUS
```

The full Python script to download tweets in real time from specific accounts is provided at this link (NB: Link to StreamAPISection_twython_keywordSample_v1.py). The key lines are:

```python
# Make function
def streamConnect(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET):
    stream = MyStreamer(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
```
The streaming API requires users’ identification number be used, not their screen name. The identification number can be downloaded via the REST API or from this website. Accounts can change their screen name, but their identification number is permanent.

3.4 Converting to Local Time

All tweets are delivered with a \texttt{created\_at} field for the date and time the tweet was created. Twitter sets this information, however, to Greenwich Mean Time, which can pose a problem for analyzes incorporating Twitter into offline events. To ensure the \texttt{created\_at} field aligns with the local time of the author of the tweet, not the time in Great Britain, the researcher must undertake a few additional steps.

If the tweet does not contain GPS coordinates, it most likely contains a field in the \texttt{user} field called \texttt{utc\_offset}. (“Most likely” because this field only exists if a user has identified his or her timezone in his or her profile; Twitter dose not guess the timezone. In my experience, 65\% of users do so.) This field gives the time, in seconds, by which the user’s self-identified timezone differs from Greenwich Mean Time. The researcher adds this value to \texttt{created\_at} to estimate the local time an account created a tweet. This new time is an estimate because it is possible the tweet author created the tweet in a different timezone than what is indicated in the profile.

If the tweet contains GPS coordinates, the local time it was created is known with certainty. With precise location information, one can identify the timezone of the tweet and adjust accordingly.

The full Python script to adjust \texttt{created\_at} to local time is here (NB: Link to Twitter\_ConvertTimeToLocalTime\_v1.py). The key lines are:

```python
1  # If tweet contains GPS coordinates
2  # Get GPS pair, SW corner of bounding box from Twitter
3  longitude = tweet['place']['bounding_box']['coordinates'][0][0][0]
```
latitude = tweet['place']['bounding_box']['coordinates'][0][0][1]

# Get timezones
tf = TimezoneFinder()
zone = tf.timezone_at(lng=longitude, lat=latitude)  # Gives string of name of timezone
timezone = pytz.timezone(zone)  # Convert string to pytz format

# Make local time
utc_time = dt.datetime.strptime(tweet['created_at'], '%a\%b\%d\%H:\%M:\%S\+0000\%Y').replace(tzinfo=pytz.UTC)  # Convert tweet timestamp to datetime object
local_time = utc_time.replace(tzinfo=pytz.utc).astimezone(timezone)  # Get local time as datetime object

### If tweets do not contain GPS coordinates
# Correct for user timezone
utc_time = dt.datetime.strptime(tweet['created_at'], '%a\%b\%d\%H:\%M:\%S\+0000\%Y')
local_time = utc_time + dt.timedelta(seconds=tweet['user']['utc_offset'])  # Subtract hours based on timezone from profile

### 3.5 Improving Location

When a tweet contains GPS coordinates, Twitter adds the location name from where the tweet originates. Most of the time - approximately 85% in my experience - this location name is the city and is correct. The rest of the time, however, the location information is instead a “point of interest” (Twitter’s phrase, apparently referring to prominent landmarks or locations), city neighborhood, or administrative region (state, in the United States). For example, a point of interest could be “888 Vietnamese Restaurant”, a location probably too precise for most analysts; reverse geocoding reveals that that restaurant is in Austin,
Texas. The reverse problem also exists: Twitter resolves many small towns to the state level, meaning an analyst can improve the precision of these points with reverse geocoding; for example, it commonly codes Druid Hills, North Carolina as only North Carolina and Vandenberg Air Force Base as California. Finally, Twitter can give neighborhood names such as Boston’s Financial District or SoMa (South of Market) in San Francisco; a reverse geocoder will reveal to which city those neighborhoods belong.

The key is to feed the tweet’s GPS coordinates to a reverse geocoding package. I prefer Python’s reverse_geocoder package. The code provided at this link (NB: Link to Twitter_AssignLocation_v1.py) shows how to do that. If Twitter indicates the place is a city, it keeps the city name Twitter assigns. If Twitter calls the place a “poi”, “neighborhood”, or “admin”, it feeds the tweet’s GPS coordinate to the reverse_geocoder package. The code preserves the place name Twitter assigns so that one can preserve the point of interest or neighborhood for future reference; it also adds county and state names. The key lines are:

```python
if (tweet[\'place\'][\'place_type\'] == \'city\'):
    tweet[\'city\'] = tweet[\'place\'][\'name\']

if (tweet[\'place\'][\'place_type\'] != \'city\'):
    # Other processing not shown here but that is in the script
    tweet[\'reversegeocode_results\'] = rg.search(tweet[\'place.bounding_box.SWcorner\']) # Perform reverse geocode
    tweet[\'city\'] = [item[\'name\'] for item in tweet[\'reversegeocode_results\']]
```

3.6 Storing Tweets

Approximately 5 million tweets per day are available to researchers via the streaming API. Uncompressed, these tweets require approximately 12 gigabytes per day. A researcher therefore needs the capacity to store these tweets and later access them quickly. Because of the
size of these data and the need to maintain a continuous connection to the streaming API, it is a good idea to have a dedicated computer with as much RAM as possible.

The easiest way to reduce the storage space required is to compress (zip) the files. A common compression protocol is gzip, and it is freely available in Unix operating systems. While the amount of space saved will vary by the kind of file, my experience is that gzipped tweets require approximately 16% the space that unzipped ones do.

To make the files even smaller, one can remove certain fields from each tweet. This subsetting can be done as the tweet arrives or once it is stored in a file. This procedure is useful because many fields in a tweet, such as the background image URL, may not be necessary depending on the project. An example of a script which reads downloaded tweets, keeps certain fields, and saves a new file is provided here (NB: Link to RESTAPISection_python_jsontoCSV.py). After removing fields I am not interested in and compressing the file, my tweets are 6.67% of their original size (the final reduction will depend on which fields and compression you use).

I do not recommend, however, removing tweets field, for three reasons. First, it throws away data, which is never a good idea. Second, it removes the flexibility of JSON formatting, which means it is brittle to changes in tweet metadata Twitter may make. For example, it does not extract image or video URLs that are now provided. Any time Twitter adds new fields to a tweet, the script will require updating. Downloading raw JSON files, one does not have to worry about changes in how the tweet is formatted. Third, tools exist to automatically convert JSON to dataframe formats. Python’s pandas library will do this, as will the csvkit toolkit. The script is thus redundant with more robust tools.

The two broad approaches to storing tweets are as flat files or in a database. “Flat file” is data in raw text form; it can be read by any text editing program like it is a normal file. A .csv file is an example of a flat file. A database is an indexed collection of files; searches for items in the files are queries of the index, increasing the speed of queries and potentially decreasing the memory load on a computer. Databases can only be explored via specialized
programs or scripts, i.e. the R command `read.csv` cannot read a database.

Within the database world are two flavors, structured and semi-structured databases. Structured data are data where each datum has the same variables. For example, all bills in a database of legislation will have values for date, sponsor, number of words, and so on. Structured data resemble the spreadsheets that most people are comfortable working with. Databases which store these data are known as SQL databases after the main language for querying them, Structured Query Language (SQL). Many kinds of data, especially from social media, do not have the same variables for each datum. For example, a tweet downloaded from Twitter will list the hashtags, links, and user mentions it contains if the tweet text has any; if none exist, an empty list is returned. Twitter will also identify any stock symbols in a tweet as long as they are proceeded by a dollar sign and are uppercase. A tweet with zero hashtags therefore looks different than one with one, and one with one looks different than one with two.

While a researcher can create a structured database that accounts for this eventuality, it is unwise to do so. First, one must define how many columns to create for the variable that may or may not exist. Returning to the hashtag example, a tweet could contain up to 47 hashtags.\(^4\) Constructing the database requires similar calculations for user mentions, stock symbols, and links. One can create a SQL database with as many columns as possible variables, but doing so leads to a much larger database than is required. Second, Twitter could decide to change the actual data a tweet contains. For example, in April 2013, Twitter added annotation to tweets if they contain stock symbols. Attempting to load a tweet with a stock symbol would crash one’s script, preventing subsequent tweets without stock symbols from entering the database. One’s script can be structured to avoid this problem, but then one would miss the data on stock symbols. Structured databases are not ideal for semi-structured data.

Semi-structured databases are designed to work with data whose representation can vary

---

\(^4\)The smallest hashtag is two characters, e.g. “#a”, and hashtags will have a space character separating them, except for the final hashtag. Solving $2x + x - 1 \leq 140$, $x = 47$. 

50
for each datum. These databases are commonly called NoSQL databases. There is not a
 dominant query language for semi-structured data; the two most common are Cassandra and
 MongoDB. Cassandra started at Facebook but is now open source, and it did not originally
 handle JSON objects (tweets are delivered as JSON objects). MongoDB is the preferred
database for JSON objects and is therefore the one most commonly used for storing tweets
in a database. (Ironically, Twitter stores tweets internally as SQL objects. See their GitHub
page for their implementation, and this Quora conversation for more information. The Social
Media and Political Participation Lab at New York University, for example, stores its tweets
in MongoDB databases. MongoDB is open source (as is Cassandra) and can be used through
R or Python.

Databases may not be necessary, however. Their advantages over flat files dominate when
the object to be scanned is too large to fit into memory or computing time is a constraining
factor, neither of which are as large an impediment as they may originally seem. First, a
day’s tweets are 12 gigabytes, and high-performance laptops commonly have 16 gigabytes of
RAM; processing and compressing the tweets after they are downloaded makes them even
smaller. Desktops commonly have 32 or 64 gigabytes of RAM, and a server much more.
Second, the connection to Twitter can be maintained in such a way as to minimize the size
of flat files. For example, my connections to Twitter’s API restart every hour, meaning I have
one file per hour’s tweets. These files are 500 megabytes raw, 33 subsetted and compressed,
and it is trivial to read a file that size into memory. The main benefit of restarting the
connection every hour, however, is that the connection intermittently drops. If the drop
occurs early in the day and the connection is maintained for the full day, then the researcher
has now downloaded little data from that day. If the connection is dropped early in the
hour, the researcher has lost only data for the rest of that hour, and the connection will be
reestablished in fewer than 60 minutes.

Though individual files may be small, one is likely to want to read very many of them
to find tweets of interest. For example, I have downloaded tweets since August 26, 2013
and frequently want to scan the $365 \times 24 \times (\text{number of days since then})$ files for tweets from a particular country. A database could do this quickly. But it is trivial to write a script to scan these files and pull the tweets that are from a country (or match any other criteria in which I am interested). How long this script takes will depend on how many files exist, how they are loaded into memory, and if the code runs parallel. My script takes a few days to scan every file, but it is rare that I need to read every file; the vast majority of files can be ignored based on their date. But computing time is inexpensive, and there are always other tasks to focus on in the meantime. Once the subset of tweets matching my criteria are found, the equivalent of the results of a database query, subsequent analysis can proceed much more quickly.

4 Clean and Analyze

This section provides scripts and examples for cleaning and analyzing tweets. Whether purchased, downloaded from the REST API, or acquired through the streaming API, raw tweets require processing in order to conduct most analyses. For example, tweets from both APIs are formatted in JSON, a convenient data structure that is not rectangular and therefore difficult to be analyzed without transformation; Sifter delivers tweets as a spreadsheet but with oddly named columns and a timestamp that needs conversion.

Section 4.1 explains how to clean data from Sifter, the REST API, and the streaming API. It explains the cleaning required and provides scripts to do that. Section 4.2 shows how to evaluate tweets for hashtags, non-hashtag keywords, retweets, user mentions, links, and language. Section 4.3 demonstrates, with an example from Bahrain, that tweets can be also used for event detection. Section 4.4 discusses analyzing data contained in links users tweet; these links can point to articles or an image the user has shared. Section 4.5 explains how to generate data outside of the context of the tweet text. Twitter delivers user metadata with a tweet, and the researcher can generate other metadata via, for example, events and
4.1 Clean

4.1.1 Tweets From Sifter

I purchased tweets from 21 accounts in Egypt and Bahrain from a three month period in early 2011; the accounts generated 55,849 tweets in that period. The raw tweets from Egypt are here (NB: Link to Tweets_Dataframe_FromSifter_Egypt.csv). Due to Twitter’s terms of service, readers interested in obtaining the raw tweets from Bahrain should contact the author. Note that the column names have been cleaned from how Sifter delivers them; this section will show how to clean the column names.

Sifter provides its data in data frame format as a comma separated file, so it is easy to load into memory. The header of the data frame is clear, but many of the fields are formatted awkwardly, are unnecessary, or both. For example, the tweet text is repeated twice, with one of the columns called “Title”, and there is also a column that shows the parameters submitted to Sifter. A majority of the columns also start with “X.M..” and end with “..”, e.g. “X.M..followers_count..”. This R script reads a data frame from Sifter, modifies the header, and writes a new file containing two fewer columns than the original (NB: Link to SwifterSection_cleanData_v1.R). This Python script does the same (NB: Link to SwifterSection_cleanData_v1.py).

The script also corrects each tweet’s timestamp. Twitter reports all times on Greenwich mean time, so the script adjusts forward two hours for Egyptian tweets, three for Bahraini ones. To do that it, it cleans how Sifter provides the timestamp. All steps were already performed on the Bahrain and Egypt datasets used later; the clean Egypt data is here (NB: Link to Tweets_Dataframe_FromSifter_Egypt_clean.csv).
4.1.2 Tweets from REST API

Since Twitter does not permit the sharing of more than 50,000 tweets per day, it may not always be possible to obtain a corpus of tweets as easily as clicking on the links in the previous section. Instead, it is more likely that you will have to download the tweets using their tweet identification numbers; Twitter allows the unlimited sharing of these. This section describes how to do that.

The process is the same as the one introduced in the scripts for downloading specific tweets from the REST API (NB: Link to RESTAPISection_twitteR_specificTweets_v1.R) (NB: Link to RESTAPISection_twython_specificTweets_v1.py). The R script at this link (NB: Link to AnalysisSection_twitteR_downloadSifterTweets_v2.R) reads in a set of of tweet IDs (saved from the raw purchased tweets and available at this link (NB: Link to Sifter_Twitter_IDs.txt)).

```r
# Empty object that tweets will feed into
tweets <- NULL
# Download tweets
for(i in 1:length(chunkedTweets)) {
  print(paste('On cycle', i, 'of', length(chunkedTweets), sep=' '))  # Status tracker
  temp <- twListToDF(lookup_statuses(ids = chunkedTweets[[i]]))
  tweets <- rbind(tweets, temp)
  print(c('Pausing'))
  Sys.sleep(delay)  # How many seconds to pause so that do not trip rate limit. Commented out in this loop because downloading 3,200 tweets will never exceed the rate limit. (60 requests * 100 tweets per request) > 3200 tweets
}
write.csv(tweets, 'Sifter_Twitter_IDS_Downloadedtweets.csv')
```

The full script steps through verifying your account with Twitter, loading the identification numbers and calculating the length of the delay based on the rate limit. Because the process is no different than downloading tweets not acquired through Sifter, I have not
created a specific Python script for this process; use the Python script for tweets based on their identification number.

Once the tweets are downloaded and into a data frame, they require further processing to integrate with the scripts developed to analyze Sifter data. This R script (NB: Link to SifterSection_cleanDownloadTweets_v1.R) modifies column names; assigns each tweet to a country; adds variables for hashtags, user mentions, retweets, and the country of the tweet author; and adjusts the time to local time. It then saves the data as “Tweets_Dataframe_Twitter_IDs_Downloadedtweets_Cleaned.csv”. This Python script does the same (NB: Link to SifterSection_cleanDownloadTweets_v1.py). The scripts in the next section will now work on either data obtained from Sifter or downloaded via the REST API and twitteR.

4.1.3 Tweets from Streaming API

Tweets downloaded from the streaming API also require some processing before they can be used for analysis. streamR preserves the raw JSON tweet when it writes files, and it converts them to a data frame with the parseTweets() function. parseTweets() does not preserve every entity (retweets, user mentions, or hashtags), however, so those need to be manually recreated. One could load the raw text file where each tweet is a JSON object, but R does not have robust packages for reading JSON data. This R script (NB: Link to AnalysisSection_cleanStreamTweets_v1.R) does that. It reads tweets saved via streamR’s sampleStream() or filterStream() functions, renames columns for consistency with other scripts in this Element, and flags tweets with hashtags, that are retweets, or that mention a user. It also formats the tweet creation time field. This Python script does the same except it reads JSON-formatted tweets (NB: Link to SifterSection_cleanDownloadTweets_v1.py).

If you elect to save the tweets in their native JSON format and want to use R, the easiest way to convert them to .csv files is to use the csvkit toolkit. csvkit is a command line utility that works on Apple or Linux machines. It has a command, csvjson, that converts a JSON
file to a .csv. You can then load that .csv with `read.csv`. In Python, you can read the JSON file with the `json` library or use the `pandas` library’s `read_json` function to read the JSON file as a dataframe.

Remember, nothing dictates that the data frames containing the tweets be formatted as laid out in the proceeding three scripts. They were done this way for consistency with each other so that subsequent analytical scripts do not have to account for the source of the tweet. Modify the scripts to your preferences.

### 4.2 Text Analysis

Twitter is most commonly used as a platform for gathering text data. This section provides code for common Twitter text analysis tasks such as counting hashtags, non-hashtag keywords, retweets, user mentions, link sharing, and language frequency. It will also show how to estimate whether tweets come from mobile or desktop devices, as well as which kind of mobile device. Tweets can be pooled across any time period; these analyses will aggregate by the hour to show average patterns in these behaviors. The Sifter data will again be used.

The following extracts are only of R scripts. Each R script adds metadata (about hashtags or links or user mentions, for example) and plots resulting trends. Graphing in R is much easier (and prettier) than in Python, so I did not replicate these scripts in Python. Instead, I created this Python script that cleans JSON-formatted tweets, adds metadata, and writes these new data as a .csv file (NB: Link to AnalysisSection_cleanStreamTweets_v1.py). It adds metadata for each situation described below. If you prefer that script to the R version, you will still need the second half of each R script to aggregate and plot the metadata.

*****

When Twitter sends a JSON formatted tweet, it has already extracted the hashtags and created a field in the tweet object that lists each hashtag. Sifter converts this information to a column where each entry is an empty string (no hashtags in the street) or a string
with a semi-colon separating each hashtag. Their processing saves the analyst a few steps. This script (NB: Link to AnalysisSection_Sifter_AnalyzeTextHashtag_v1.R) creates variables to count tweets with at least one hashtag as well as the number of hashtags in each tweet. It then aggregates by country-hour and generates four plots, which are shown in Figures 1-4. The key lines are:

```r
1 data$hashtag_dummy <- ifelse(nchar(data$hashtag) == 0, 0, 1)
2 data$hashtag_count <- apply(as.matrix(data$hashtag), MAR=1, FUN = function(x)
                           ifelse(nchar(x) == 0, 0, length(unlist(strsplit(x, ';'))))) # If the hashtag column has length 0, the tweet contains 0 hashtags. If the length is not 0, count the number of hashtags by splitting the string on the semi-colon; that is where the split occurs because Sifter uses the semi-color to separate hashtags. apply() does this to each tweet.
3 data_hashtagAgg <- as.data.frame(data %>% group_by(Country, Hour) %>% summarize(tweet_with_hashtag = sum(hashtag_dummy),
                           tweet_with_hashtag_perc = sum(hashtag_dummy)/sum(count),
                           hashtags_per_tweet = mean(hashtag_count),
                           conditional_hashtags_per_tweet = mean(hashtag_count[hashtag_dummy==1])))
```

Because a hashtag is simply a word with a symbol prefix, searching a tweet for a particular keyword entails a similar process. Finding a particular keyword is slightly more difficult because neither Twitter nor Sifter process the tweet text for keywords. Finding a particular keyword is useful if you have been collecting a random sample of tweets and become interested in a particular event or term after the tweets are downloaded. Note that it provides code for English keywords only. The process is the same for non-English tweets, so long as one knows the UTF-8 code for the characters that comprise the terms in which you are interested. For more on UTF-8 and R, see R’s encoding function, documentation on searching inside a string, and these examples.

The script to find keywords is here (NB: Link to AnalysisSection_Sifter_AnalyzeTextKeywords_v1.R). These lines show how to search a tweet for a specific keyword:

```r
1 data$protest <- ifelse(grepl('protest', data$tweet_text, ignore.case = TRUE)
```

57
Figure 2: Percent of Tweets with 1 or More Hashtags

- Bahrain
- Egypt
Figure 3: Hashtags per Tweet

![Chart showing hashtags per tweet for Bahrain and Egypt over different hours.](chart.png)
Figure 4: Hashtags per Tweet with Hashtag
Note that this will find when that word is a hashtag ("Let’s go #protest") and stands on its own ("Let’s go protest"). The script then aggregates and plots these words. Figure 5 shows the result.

Finding a tweet with a link is a similar process to finding a keyword. Instead of looking for a whole word, however, one can look for “http://”. The key line then becomes:

```r
# If a tweet contains http, assume that is a link. Case insensitive.
```

This script (NB: Link to AnalysisSection_Sifter_AnalyzeLinks_v1.R) contains the code to identify links and generate a plot of the percent of tweets with links by country-hour. That output is shown in Figure 6.

Finding retweets should be done two ways. First, Twitter identifies a retweet with the retweeted key in the JSON formatted tweet. Sifter translates that to the “is_retweet” column with Boolean flags; the script to clean Sifter data converts those flags into a dummy variable. Second, one should read each tweet for the “RT” characters, paying attention to
Figure 5: Keywords

![Graph showing trends for keywords and countries over hours with various colored lines for different keywords and countries.](Image)

Keyword
- Protest
- Police
- jan25
- feb14

Country
- Bahrain
- Egypt

Percent of Tweets

Hour
Figure 6: Percent of Tweets with Links
capitalization. Twitter only identifies a retweet if a user has used Twitter’s interface to retweet a tweet (a “native retweet”), but users sometimes retweet by typing “RT @<screen name of tweet creator>” and pasting the copied tweet. Since Twitter does not flag these tweets as retweets, a manual check ensures all retweets are found. In the Sifter sample, only 501 tweets are native retweets, but 12,717 contain “RT ”. Figure 7 shows this difference, and the full script for identifying retweets is at this link (NB: Link to AnalysisSection_Sifter_AnalyzeRetweets_v1.R). The key line is this:

```r
1 data$retweet_manual_ignoreCase <- ifelse(grepl('RT@', data$tweet_text, ignore.case = TRUE) == TRUE, 1, 0)
```

Like hashtags, Twitter treats user mentions as an “entity”, meaning it extracts the screen names of those a tweet mentions before delivering that tweet via its API. Twitter’s processing saves the researcher from using regular expressions, facilitating analyzing the data. Sifter takes the extract screen names and creates a column called “user_mention_username” where each value is one string containing each username mentioned in the tweet. Processing these data is similar to finding hashtags. The script for user mentions is here (NB: Link to AnalysisSection_Sifter_AnalyzeTextUserMentions_v1.R), and the key lines are:

```r
1 data$mention_dummy <- ifelse(nchar(data$user_mention_username) == 0, 0, 1)
2 data$mention_count <- apply(as.matrix(data$user_mention_username), MAR=1, FUN = function(x) ifelse(nchar(x) == 0, 0, length(unlist(strsplit(x, ';')))))
# If the Sifter mention column has length 0, the tweet contains 0 user mentions. If the length is not 0, count the number of mentions by splitting the string on the semi-colon; that is where the split occurs because Sifter uses the semi-color to separate user mentions. apply() does this to each tweet.

3 # Specific user mentioned? Use @ZacharyST as example
4 sum(grepl('ZacharyST', data$tweet_text, ignore.case = TRUE)) # 0.
```

I leave it to the reader to create the user mention figures. I will provide the figures upon
Figure 7: Percent of Tweets that are Retweets

- **Retweet Method**
  - Native Retweet
  - Read Tweet for ‘RT’

- **Country**
  - Bahrain
  - Egypt

The graph shows the percentage of tweets that are retweets over different hours of the day for Bahrain and Egypt. The retweet methods include native retweets and reading the tweet with the ‘RT’ phrase.
request, if you would like to check your work.

Twitter also labels the language a tweet is written in, using the language’s two letter ISO-639-1 code. Sifter, however, does not convey this information, so one has to recreate it. In R, the best way is through the textcat package, which uses the character based n-gram methodology developed in Cavnar and Trenkle (1994). The script to assign language is here (NB: Link to AnalysisSection_Sifter_AnalyzeLanguage_v1.R), and the key lines are:

```r
1 data$language <- apply(as.matrix(data$tweet_text), MARGIN = 2, FUN = function(x) textcat(x))
2 3 data$language[is.na(data$language)] <- 'arabic' # Convert the NA to Arabic
4 5 data$english <- as.numeric(ifelse(data$language == 'english', 1, 0))
6 data$arabic <- as.numeric(ifelse(data$language != 'english', 1, 0)) # Assumes all non-English tweets are Arabic
```

Figure 8 shows the distribution of languages by country-hour. Note that textcat() does not work well on Arabic text and confuses many English tweets for Gaelic or Scottish English. To account for this misassignment, the script manually recategorizes some languages. It also puts most of the languages into an “Other” category, to facilitate visualization.

Finally, Twitter provides metadata on the source of the tweet. The metadata is a string such as “http://www.twitter.com” or “Twitter for iPhone”, and Sifter rewrites those strings to be clearer. Table 2 shows the distribution of these sources in the Sifter dataset, and Figure 9 shows the distribution of these sources by country-hour. Note that I have categorized the sources by whether or not they are mobile devices or not. The code to generate the table and figure is here (NB: Link to AnalysisSection_Sifter_AnalyzeMobility_v1.R), and the key lines are:
Figure 8: Percent of Tweets by Language

![Graph showing the percentage of tweets by language over time for different countries and languages.](image-url)
```r
xtable(as.table(sort(table(data$source), decreasing = TRUE)))  # Outputs latex table

desktop <- c('web', 'Choqok', 'TweetDeck', 'HootSuite', 'Ping.fm')  # The sources most likely to be from a desktop computer

data$desktop <- ifelse(data$source %in% desktop, 1, 0)
data$mobile <- ifelse(!(data$source %in% desktop), 1, 0)
```
Table 2: Frequency of Tweet Sources in Activist Data

<table>
<thead>
<tr>
<th>Tweet Source</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choqok</td>
<td>22285</td>
</tr>
<tr>
<td>web</td>
<td>17424</td>
</tr>
<tr>
<td>Twitter for BlackBerry</td>
<td>5142</td>
</tr>
<tr>
<td>Twitter for iPhone</td>
<td>4157</td>
</tr>
<tr>
<td>Gravity</td>
<td>2003</td>
</tr>
<tr>
<td>berSocial</td>
<td>1977</td>
</tr>
<tr>
<td>Facebook</td>
<td>741</td>
</tr>
<tr>
<td>HootSuite</td>
<td>614</td>
</tr>
<tr>
<td>TweetDeck</td>
<td>353</td>
</tr>
<tr>
<td>Snaptu</td>
<td>344</td>
</tr>
<tr>
<td>twitterfeed</td>
<td>282</td>
</tr>
<tr>
<td>Ping.fm</td>
<td>178</td>
</tr>
<tr>
<td>Tweet Button</td>
<td>148</td>
</tr>
<tr>
<td>Samsung Mobile</td>
<td>45</td>
</tr>
<tr>
<td>Twitter for iPad</td>
<td>32</td>
</tr>
<tr>
<td>Google</td>
<td>27</td>
</tr>
<tr>
<td>oauth:173069</td>
<td>21</td>
</tr>
<tr>
<td>Mobile Web</td>
<td>17</td>
</tr>
<tr>
<td>harassmap.org</td>
<td>13</td>
</tr>
<tr>
<td>Bambuser</td>
<td>11</td>
</tr>
<tr>
<td>TwitLonger Beta</td>
<td>10</td>
</tr>
<tr>
<td>Yfrog</td>
<td>8</td>
</tr>
<tr>
<td>TweetMeme</td>
<td>5</td>
</tr>
<tr>
<td>Twitpic</td>
<td>4</td>
</tr>
<tr>
<td>The BOBs</td>
<td>3</td>
</tr>
<tr>
<td>See Wh0 Viewed Y0ur Pr0file</td>
<td>2</td>
</tr>
<tr>
<td>My Tweet Lovers</td>
<td>1</td>
</tr>
<tr>
<td>oauth:3294</td>
<td>1</td>
</tr>
<tr>
<td>StumbleUpon iPhone</td>
<td>1</td>
</tr>
</tbody>
</table>
Figure 9: Percent of Tweets by Country-Source

The graph shows the percentage of tweets by country and source over different hours of the day. It distinguishes between Bahrain and Egypt for country and Desktop and Mobile for source. The x-axis represents the hour of the day, while the y-axis shows the percent of tweets. The lines for each category (country and source) fluctuate throughout the hours, indicating variability in tweet activity.
4.3 Event Detection

Twitter can also be used to generate events data. In the field of political conflict, scholars of subnational conflict rely on events data for theory testing. Events data are datasets that use sources such as non-governmental organization reports (Davenport and Ball, 2002), military records (Weidmann, 2014), or, most commonly, newspaper articles (Metternich et al., 2013; Weidmann and Ward, 2010) to record political interaction between two actors. These sources have known common problems, however, from poor geographic resolution (Hammond and Weidmann, 2014), urban bias (Weidmann, 2014), a preference for unexpected novelty, and, for newspapers, bias towards local elites who can interlocute with reporters (Kalyvas, 2004).

The current best practice uses machine coding of newspaper articles to measure daily events; examples include the GDELT, Phoenix (its open source improvement), and the ICEWS (Leetaru and Schrodt, 2013; Analytics, 2015; Boschee et al., 2015). These datasets are a significant improvement over previous work because they are updated daily and can draw on one to two orders of magnitude more data sources. They still suffer, however, the same bias as hand-coded datasets. Analyses based on these databases are therefore still likely to underreport ongoing events (Masad, 2013) or those outside of cities (Weidmann, 2014).

Tweets complement the current state-of-the-art in four ways. First, focusing on individual accounts means one should observe events that would not otherwise appear in events data. Second, these events should exhibit less of an urban bias. Third, even with cities, they allow interested parties to see events with a greater geographic resolution, e.g. a tweet may report on a march in a specific neighborhood or an attack on a police station. For example, events at the city-level, e.g. a protest in Central Park and another in Wall Street would count as one protest in existing datasets but two in a Twitter events dataset. Fourth, the decay in interest may result in a smaller decay in signal: so long as these events are relevant to some people, they should appear in a dataset for longer than in datasets that rely on newspapers.

To explore this possibility, I worked with two students to generate events data from a
subset of the tweets provided in the Bahrain Sifter tweets (provided upon request). Two undergraduates from the University of California, San Diego have manually coded tweets from human rights activists in Bahrain spanning February 14th, 2011 through March 17th of the same year, the height of Arab Spring protests in Bahrain. I then aggregated and deduplicated their coding in order to compare the Twitter results to ICEWS; the ICEWS data are contained at this link (NB: Link to events.2011.20150313084656.tab).

Figure 10 shows that events from these tweets reveal much different patterns than ICEWS, the dataset to which I compared them. The tweets record events across many more locations than ICEWS, an average of 20 per day versus ICEWS’ 3. For example, the tweets record clashes in suburbs like Duraz or outlying cities like Sanabis, and there are two reports of nerve gas used against protestors. Within Manama, clashes are recorded at the airport, Dana Mall, and Bahrain University, among other places too precise for ICEWS to reference. Other locations include an activists home and Sitra, Bahrain’s seventh largest city. The tweets also record more actors than ICEWS, an average of 25 versus ICEWS’ 12. The tweets record a similar number of events as ICEWS, though it does record more events at the start of Bahrain’s protests.

In addition to recording more locations and actors than ICEWS, the tweets events data record different patterns of activity. ICEWS records a spike in actors at the beginning and end of the period contained in the tweets, with a lull in the middle (late February to mid-March). The tweets record the same pattern, but with greater spikes at the beginning and end of the sample period; from late February to mid-March, the tweets record the same number of actors as ICEWS. Most interestingly, ICEWS records the same number of locations during the sample period. The tweets, on the other hand, show a u-shape: events happen in more locations at the beginning and end of the protest period, with few locations recording events from late February to early March. In other words, ICEWS appears to record events in the same areas over time. Overall, events data from Twitter appear to report more actors engaging in more activity across more location than current best practice
data suggest.

The code to generate events data and the figures is here (NB: Link to GetEventDataFromUndergradCoding_v1.R), and the handcoded datasets are here (NB:Link to Coder1_Tweets_Dataframe_FromSifter_AdjustedTime.xlsx) and here (NB:Link to Coder2_Tweets_Dataframe_FromSifter_AdjustedTime.xlsx). The code is meant purely as a proof of concept. Its geocoding relies on human-identified places in tweets, and the same place is often spelled multiple ways. Only English tweets were coded. The limits of English compared to local language tweets is probably real, but it is unknown; they are certainly not as limited as English newspapers. Most importantly, a real-time dataset that records multiple kinds of events across the globe and with intracity resolution cannot be created by humans. An eventual system would need to be fully automated, and this script does not start the learning process that would create a classifier for a final product.
Figure 10: Twitter Provides Events Data

(a) Events

(b) Actors

(c) Locations
4.4 Links and Imagery

It is common for users to share links in their tweets. The content of links is a potentially rich source of information for researchers, but they remain understudied. The most common way links are analyzed is for researchers to note the percentage of tweets containing them (Suh et al., 2010; Steinert-Threlkeld, 2016), but I am aware of no articles that analyze the content of those links. There are two reasons links remain understudied.

First, spam accounts tweet links. The presence of a link, in conjunction with age of an account and the use of trending topics, is a common way researchers identify spam accounts (Kwak et al., 2010). While bots may represent for 6% to 8.4% of all accounts, it is unknown what percent of links they share (Lotan et al., 2011); it is probably more. (Not all bots are spam accounts, but I am not aware of any work which manually identifies spam accounts.) Astroturf political campaigns commonly use authentic looking accounts controlled by a political operation to share one or a few links, creating the appearance of grassroots concern where none exists (Mustafaraj and Metaxas, 2010; Ratkiewicz et al., 2011). Without a reliable, precise spam filter, a researcher studying links risks studying spam.

Second, studying links requires additional processing work. When Twitter delivers a tweet with a link, it extracts the link for the downloader. It does not, however, deliver the content contained at the link. Theoretically, one could estimate the link’s content by reading the URL, as newspaper links often contain the headline. Twitter, however, automatically shortens links; while useful to the tweet creator, the shortening means information the full URL contains is removed, and the researcher has to follow the link to the webpage. A researcher interested in link content therefore has to build a web crawling system on top of the one connecting to Twitter. Moreover, many of these links are likely to be spam and not interesting to a social scientist. Finally, there exists no standard method of presenting content on the internet. It is relatively easy to build a website crawler for one or a few sites, as a website will deliver its content consistently. Building a crawler to robustly account for
all possibilities is much more complicated.

The most compelling reason to study links is because they often point to images, and image analysis is a new frontier of large-scale analytics. Twitter is traditionally text focused, but imagery is becoming the most popular content shared on social networks. Images are commonly shared on Twitter, including images containing more than 140 characters of text. I am aware of no study which quantifies what percent of tweets contain imagery, though a few studies have analyzed images on Twitter. Kaneko et al. (2013) use geo-tagged tweets with photos to detect sporting events and even fireworks. Mehrdad Yazdani and Lev Manovich correlate 1 million images from 20 United States cities over one year with self-reported measures of happiness as well as socioeconomic indicators (Yazdani and Manovich, 2015).

Image analysis for the social sciences is still in its infancy, however, because computational requirements for images are higher. Image files are much larger than text files, and feature detection is more complicated. It is relatively easy to create variables (features) from text, but determining which parts of an image file represent quantities of interest (people or places, for example) is much harder. Recent advances in neural networks, colloquially now called “deep learning”, are promising, but these models require extremely powerful computers and generate classifications in ways that make the features determining those classifications difficult to understand. If a tweet contains an image, Twitter provides the link to the image but not the image itself; the researcher must then add this processing step to the data acquisition pipeline. I am aware of only one paper which uses images to answer a political question, and it is still a working paper (Anastasopoulos et al., 2016).

Satellite imagery is a promising subset of image analysis and the one most likely to be incorporated first in analyses. Images of light have already been used to measure economic development, for example. Because Earth’s colors and geographic features are more stable than random photos from a social network, the first rigorous use of image data in the social sciences will probably come from satellite imagery. Currently, these images are best for cross-sectional analysis, as satellites pass over places irregularly.
4.5 Enriching Tweets

One can enrich the information in a tweet’s text four ways. If conducting text analysis, any of these approaches enable a structural topic model (Roberts et al., 2014).

First, one can consult the metadata Twitter provides with a tweet. These data include how many followers the author has, how many accounts the author follows, when the account was created, the account’s default language, the language of the tweet, and if the tweet has GPS coordinates, among others. The user’s self-reported location is provided in approximately 50% of tweets (Leetaru and Schrödt, 2013), and individuals commonly make their screen name the same as, or similar to, their real name (Barberá, Jost, Nagler, Tucker and Bonneau, 2015).

Second, one could also interview the individuals behind an account. Two difficulties arise. First, messaging an account requires that the recipient follows the account in question. In other words, survey respondents have to first follow the survey administrator, which is unrealistic. Second, interviewing would require IRB approval. A more expeditious approach is to interview people and ask if they use Twitter (Tufekci and Wilson, 2012; Zickuhr, 2013).

Third, accounts’ tweeting patterns and social network can reveal information not in tweets’ text. For example, accounts belonging to unemployed individuals tweet more during the day, and cities with higher levels of unemployment have low communication entropy (Llorente et al., 2014). The style of tweets gives some indication of an account’s age (Nguyen et al., 2013). Social class, ethnicity, and education can be estimated probabilistically when tweets contain GPS coordinates (Malik, Nakos, Lamba and Pfeffer, 2015). A user’s social network is also more predictive of that user’s age and political affiliation than relying on just user attributes (Zamal, Liu and Ruths, 2012b). These techniques are compelling, but there exists no R or Python packages that implements them automatically, increasing the costs of their use. A researcher may have to reinvent the wheel each time; the need for such data should be great before time is invested in this approach.

Finally, one can manually inspect each account in a sample. By viewing a profile’s photo,
gender may be obvious, and an age range could be created. An account’s past tweets can give an indication of the author’s primary location and interests. Googling the account name may reveal other sites where the author has registered, and those sites may provide more demographic information. This manual approach is what Driscoll and Steinert-Threlkeld (2016) use to assign occupation, age, and gender to the accounts in their sample (Driscoll and Steinert-Threlkeld, 2017). To create data on enough accounts for a quantitative study using this approach requires cheap human labor.

Another enrichment involves connecting aggregated Twitter data to other datasets. For example, Twitter users can sometimes be found in voter files or campaign donation contribution datasets; this approach has been used to validate political inferences made from individual’s network on there (Barbera, 2015). Inferences can be made about tweets aggregated to census tract levels, both in the United States (Mislove et al., 2011; Malik, Lamba, Nakos and Pfeffer, 2015) and elsewhere (Sloan et al., 2015). Tweets aggregated to the country level can also be connected to events datasets (Steinert-Threlkeld et al., 2015; Steinert-Threlkeld, 2016).

Other datasets entice. With the recent release of IRS 990 filings, it may be possible to link Twitter accounts to charitable donations to non-profit organizations. For example, it would be interesting to see if the ideal points of non-profit organizations correlates with the ideal points of their donors. While reliable census data tends to come from wealthy countries, countries such as the Philippines, Turkey, and Ukraine have disaggregated census data that have not been combined with Twitter data.

5 Twitter in the Social Sciences

This section provides examples of applications other social scientists have made of Twitter data. It also summarizes competitors to the platform — Facebook, Reddit, Sina Weibo, Tumblr, and Instagram — as well as the potential of call detail record data.
5.1 Research on Social Media

The speed of academic publishing in the social sciences means that papers taking advantage of Twitter data have only started to appear after 2010 (Twitter was founded in 2006). The appeal of Twitter, and “big data” more broadly, is that it provides data on more people in more places across more time than scholars could realistically hope to achieve with survey methods. For example, it has long been known that individuals’ happiness is lowest during the middle of the day, and lower during the week than on weekends; the cost of acquiring data meant these results were only tested in WEIRD (Western, educated, industrialized, rich, and democratic) countries (Henrich, Heine and Norenzayan, 2010), but Twitter reveals that it applies in at least 84 countries (Golder and Macy, 2011). Countries also experience similar changes in their total happiness, though baseline levels vary (Poblete et al., 2011). Because people use social media to talk about topics like health, these platforms can also be used to monitor public health and identify individuals susceptible to treatment (Charles-Smith et al., 2015).

In political science, Twitter has been used in two main areas of research: conflict dynamics and public opinion. For an early review of social media and social movements, see the 2013 special issue of American Behavioral Scientist called “New Media and Social Unrest” (Tufekci and Freelon, 2013). For a review of the the potentials and challenges of using online social network platforms for research, see Golder and Macy (2014). For a series of essays on the role of big data in the social sciences, see the 2015 symposium in PS: Political Science and Politics titled “Big Data, Causal Inference, and Formal Theory: Contradictory Trends in Political Science?”

Thomas Zeitzoff has three papers which show how Twitter can generate new insights about conflict and foreign policy. Because the costs of posting on Twitter are much lower than for publishing in a newspaper or broadcasting on television, individuals and small organizations have become sources of events data. Zeitzoff (2011) combines @AJGaza (an Al-Jazeera Twitter account) and @QassamCount (records rocket attacks into Israel) tweets
with blog reports and a Wikipedia event timeline to examine the microdynamics of Israel’s 2009 war in Gaza, finding that:

Hamas’s and Israel’s response intensity double immediately after the introduction of ground troops and that immediately following the UN Security Council vote, Israel cuts its response intensity in half, while Hamas slightly increases. (939)

This paper appears to be the earliest incorporation of social media data into conflict studies.

Aday et al. (2012), in a report for the United States Institute of Peace (USIP), present one of the earliest usages of Twitter in the social sciences. In a similar report for the USIP from 2010, many of the same authors proposed a framework for analyzing the effect of online media on contentious politics. Their data, however, was limited to a case study of digital media during Iran’s 2009 protests (Aday et al., 2010). They examine the geographic distribution of individuals clicking on bit.ly links in tweets and find that:

new media informs international audiences and mainstream media reporting rather than plays a direct role in organizing protests or allowing local audiences to share self-generated news directly with one another. [...] The key role of new media may be its bridging function: from an activist core to mass publics, from user-generated content to mainstream mass media, and from local struggles to international attention. (5)

Later work uses Twitter to study protest mobilization directly. Combining geolocated tweets with computer-generated events data, scholars have found that the more that peripheral members of a country’s social network tweet about protests, the more subsequent protests there are (Steinert-Threlkeld et al., 2015; Steinert-Threlkeld, 2017b). A study comparing the Twitter network of people who protested in Paris after the Charlie Hebdo attacks with those who did not, finding that protestors and non-protestors segregate on Twitter (Larson et al., 2016).

Twitter also reveals how social networks vary by country. Zeitzoff et al. (2015) show how follower relationships on Twitter provide social network data that can be used to infer issue
salience (Zeitzoff, Kelly and Lotan, 2015). Combining users tweeting in English, Farsi, and Arabic with network analysis of Farsi and Arabic blogs, the authors show that users separate into identifiable ideological clusters but that network topology varies depending on which set of users is considered. Twitter can also reveal daily changes in the social network of its users. In Egypt and Bahrain, over a three month period in early 2011, the two countries’ Twitter networks become more similar and a multitude of communities arise which bridge communities in both countries (Steinert-Threlkeld, 2017a).

Not only can Twitter (and social media more broadly) provide data on events in a conflict, it may influence conflicts directly. The percentage of protestors at Egypt’s Tahrir Square who heard about the protests on social media was much higher than the percentage of Egyptians on Twitter (Tufekci and Wilson, 2012). In Russia’s 2011 parliamentary elections, users of Twitter and Facebook were more likely to believe electoral fraud occurred, while users of indigenous social network sites did not have that perception (Reuter and Szakonyi, 2013); the direction of causality is unclear, however. In Ukraine, early participants in the Euromaidan protests were much more likely to have learned about the protests via Facebook than later ones (Onuch, 2015). See Bastos et al. (2015) for an analysis of Twitter and Facebook during the Occupy, May 15, and Vinegar protest movements. In Israel’s 2012 conflict with Gaza, both sides varied their attack intensity based on international public opinion, where public opinion is measured as the number of times a pro-Israel or pro-Hamas hashtag is used per hour (Zeitzo, 2014). While a potential causal role for social media is still hotly debated, there now exists a critical mass of work in this domain that suggests further theoretical and empirical development is warranted (Tucker et al., 2016). For an example of Twitter being used to exogenously induce mobilization (in support for an online petition), see Coppock et al. (2016).

Another domain in which social media and Twitter have allowed scholars to make advances is measuring political preferences. For example, many scholars worry that digital communications technology allows individuals to consume information which reinforces al-
ready held opinions (Adamic and Glance, 2005; Farrell, 2012). Early work using Twitter finds that retweet networks sort on ideological preference while user mention networks do not (Conover et al., 2011, 2012). A limitation of this work, however, is that ideology is manually inferred based on the topic of the tweet, making the results difficult to scale. A more recent advance is to observe which political accounts Twitter users follow; these relationships place individuals in latent political space that closely matches what DW-NOMINATE would assign, making it possible to estimate millions of individuals’ otherwise unobservable political preferences (Barbera, 2015).

Research using this method has found evidence of ideological exchange: while polarization exists, it varies by topic, and ideological communities are not hermetic. It appears that, in the United States, the less political a topic is, such as a sporting event or entertainment news, the more news flows across ideological boundaries; even for political news, 12% - 60% of retweets are from users on opposite sides of the political spectrum (Barberá, Jost, Nagler, Tucker and Bonneau, 2015). A study of the United States, Spain, and Germany makes an even stronger claim: 33% - 45% of the accounts Twitter users follow are from the opposite side of the political spectrum, and individuals become more moderate over time as their network grows political diverse (Barberá, 2014). On Facebook, individuals who report whether they are liberal or conservative have approximately 20% of their friends from across the aisle, and 22% - 30% of the news they see contravenes their politics (Bakshy, Messing and Adamic, 2015). Interestingly, Barbera (2015) finds that conservatives have fewer cross-cutting ties than liberals, while Bakshy et al. (2015) find the opposite.

Because Twitter is popular with politicians, it also provides insight into how they and their constituents interact. Early work looking at how Congress people use Twitter finds that male Republican House members tweet the most, with Senators tweeting less than Representatives and those in office longer tweeting more; for all members of Congress, tweets are used to broadcast policy positions but not rally the public to action (Hemphill, Otterbacher and Shapiro, 2013). When running for office, female candidates have more followers
are tweet more than male candidates, while both major parties use the platform similarly (Evans, Cordova and Sipole, 2014). Later work uses the 354,860 tweets every member of Congress sent between January 1st, 2013 and March 15, 2014, their follower network, and a random sample of the tweets of those followers to analyze this interaction in more detail (Barberá et al., 2014). This work finds that legislators are more responsive to their most partisan constituents (though Democrats also respond to non-Democrat constituents), while constituents do not change the topics of their tweets based on legislators’ tweets.

A cottage industry exists that uses Twitter to predict voting and stock market outcomes. The title of one of the earliest papers is explanatory enough: “Twitter mood predicts the stock market” (Bollen, Mao and Zeng, 2011); using tweets from ten months in 2008, this paper finds that some aggregate measures of mood correlates with changes in the Dow Jones Industrial Average up to six days in the future. Later work finds similar correlation at an hourly level (Zheludev, Smith and Aste, 2014). Researchers have found that the number of mentions of political parties correlates positively with those parties’ electoral success in Germany (Tumasjan et al., 2010). For an exhaustingly thorough catalogue of how Twitter has been used to study politics, see Jungherr (2014).

Twitter also provides “polling” for places otherwise difficult survey, especially when longitudinal analysis is required. For example, following certain users can allow scholars to construct panel data that may reveal changes in a society’s beliefs. In Egypt, 17 million tweets from 7,000 users in Egypt have been used to measure discourse polarization there, and polarization on Twitter increased before violent protests in November 2012 (Weber, Garimella and Batayneh, 2013). Borge-Holthoefer et al. (2015) analyze 6 million Arabic tweets from Egypt from the middle of 2013. With data on 120,000 users and their 3,200 previous tweets, a classifier for pro- or anti-military tweets, and known secular or Islamist accounts, the authors show that few users, never more than 3% on any given day, express views that contradict their previous preferences as expressed on Twitter; no side gains more adherents from this switching (Borge-Holthoefer et al., 2015). In Ukraine, Twitter can measure
pro-Ukraine and pro-Russia sentiment across the country from the start of that country’s protests through its civil war; because Twitter polling does not require enumerators, it is especially useful in violent areas of the world such as Ukraine’s Luhansk and Donetsk oblasts (Driscoll and Steinert-Threlkeld, 2017).

The failure of voter turnout models used to predict the United States’ 2016 presidential election suggests that many voters are also difficult to survey. If the election suggests that traditional institutions such as the political party and media are not as influential over individuals as previously believed, then data sources which provide direct access to those individuals may become an important source of polling. President Donald Trump’s personal use of Twitter also defied expectations and galvanized new political actors. That these actors may misrepresent themselves to pollers, if they respond in the first place, but may exhibit more candor online suggests that Twitter may have more relevance for understanding American political behavior than previously thought.

The presidential election also saw the first use of “fake news” on social media platforms, especially Facebook. (“Fake news” refers to articles that appear to detail actual events but are designed instead to generate internet traffic against which advertisements can be sold. They generate traffic with sensationalist headlines.) Because the behavior is so novel, there does not yet exist a method for automatically detecting fake news; in response to post-election backlash, Facebook’s solution is to flag questionable articles for independent organizations to verify manually. While one’s initial desire may be to remove tweets containing fake news from a dataset, they should be preserved. Their presence presents one with the opportunity to study information flows in all its forms. Just as social media may provide insight into difficult to survey individuals, it may also let us study types of information flows previously unobservable. Before, the only fake news was in tabloids and rumors; now, that fake news submerges itself in an observable, recordable, and measurable medium. Do fake news items spread differently than true ones? Are they more, less, or as likely to spread as true news items? If they spread, do they stay within homogenous political communities? We will not
know how fake news works if we ignore it altogether.

It should be noted that much skepticism exists concerning the ability of Twitter, and social media more broadly, to serve as a polling platform (Tufekci, 2014). While Twitter has 300 million global monthly active users and is used by all demographics in the United States, Pew Research has found its users skew towards the young and minorities (Duggan, Page and Manager, 2016), while users who tag their tweets with GPS coordinates tend to be higher income, urban, or minorities (Malik, Lamba, Nakos and Pfeffer, 2015). Papers which claim to predict an outcome tend to conflate correlation with prediction, make post hoc claims, and half of the Twitter studies conducted before 2013 failed to predict correctly election outcomes (Gayo-Avello, 2013). These shortcomings may reflect the novelty of working with these data, as no agreed-upon methodology exists for selecting users, cleaning tweets, measuring sentiment, or measuring prediction accuracy (Gayo-Avello, 2013). Future work needs to reorient its claims, reweight its samples, and gain external validity by using panel, instead of cross-sectional, data (Diaz et al., 2014).

Finally, Twitter is especially exciting for the study of social networks and mobilization. For example, voter mobilization studies interested in social networks rely on surveys to measure the effect of social networks on voting (Lake and Huckfeldt, 1998; Dalton et al., 2002; Nickerson, 2008). While Nickerson (2008) is able to exogenously assign treatment, most work relies on respondents reporting if their friends vote or share their political preferences. While I am aware of no published research which uses Twitter to mobilize voters, it has been used to mobilize support for online petitions (Coppock, Guess and Ternovski, 2016). Facebook was used in the 2010 United States Congressional Election to mobilize 340,000 additional voters (Bond et al., 2012). Studies of mobilization in the context of the United States Civil Rights movement (McAdam, 1986) or East Germany’s 1989 protests (Opp and Gern, 1993) are similarly reliant on post hoc observational survey data. Twitter allows scholars to monitor protests in real time, allowing scholars to observe for the first time recruitment as it happens (González-Bailón et al., 2011), the differential effects of individuals in the core and
periphery of a country’s social network (Barberá, Wang, Bonneau, Jost, Nagler, Tucker and González-Bailón, 2015; Steinert-Threlkeld, 2016), and how social network structure affects mobilization (Steinert-Threlkeld, 2016; Larson et al., 2016).

5.2 Competitors

Twitter is unique in its global reach and data availability. This combination means it is probably the most studied social network. As of September 21st, 2016, a Google Scholar search for “twitter” returns 6,370,000 items, “facebook” 5,390,000, and “instagram” 172,000. Other common platforms are Tumblr (owned by Yahoo), reddit, and Sina Weibo. For a history of social networks and the internet, see Bury et al. (2013).

Founded in 2004 for elite U.S. undergraduates, Facebook opened itself to anyone over the age of 13 on September 26, 2006. It is now a major corporation with global market penetration; with over 1 billion users, it is the most popular social network and one of the internet’s most visited sites (Bhatia, 2016; Solon, 2016). Facebook is the first social network platform to reach such a large audience, and it quickly drew attention from academics of all disciplines. The first article about it, discussing privacy concerns, appeared in 2005 and analyzed information sharing behaviors of college students (Jones and Soltren, 2005). The ability to observe social connections across large groups of people has drawn the interest of network scientists (Lazer et al., 2009; Gjoka, Irvine and Butts, 2010; Ferrara, 2012), physicists and computer scientists (Catanese et al., 2011; Ugander et al., 2011), social scientists (Lewis et al., 2008; Bond et al., 2012; Reich, Subrahmanyan and Espinoza, 2012), and humanists (Rahimi, 2011). For a review of Facebook studies in the social sciences, see Wilson et al. (2012).

Facebook is the juggernaut social network and maintains a well-documented API. The difficulty for researchers is that the vast majority of accounts are private. Facebook initially maintained an internal research team that worked with academics on a broad range of questions, but after an international outcry over one study that showed some users negative,
and others positive, news (Kramer, Guillory and Hancock, 2014), it has become much more cautious in its academic partnerships. Its internal research is now more focused on selling advertisements than it was before that study. Perhaps as part of that turn, only media publishers can access the feed API (the equivalent of Twitter’s streaming API). The graph API, which gives information on friendships, is still available.

Reddit, founded in 2005 and now visited by 234 million unique individuals per month, is perhaps the Web 2.0 platform which least resembles a social network. “The front page of the internet”, Reddit revolves around subreddits, a thematic collection of user-provided material (e.g. photos, questions, essays, links to other stories) that anyone can create or follow. Users vote these material up or down and subscribe to subreddits, but one does not connect to other accounts like on a social network. Scholars have paid less attention to the site, with the studies that do exist focusing on attention dynamics (Bergstrom, 2011; Gilbert, 2013; Lakkaraju, McAuley and Leskovec, 2013).

Sina Weibo is the most similar service to Twitter, but it is targeted at users in China. Introduced on August 14, 2009 to replace the recently banned Twitter and Facebook, it has slightly over 200 million active monthly users posting 100 million messages per day. Though open to anyone, only Chinese citizens use it regularly, and there is some concern that its users are fleeing to messaging applications like WeChat and WhatsApp. Like with Twitter, many studies focus on event analysis (Qu et al., 2011; Guan et al., 2014), its use during crisis events (Guan et al., 2014), user behavior (Gao et al., 2012; Yu, Asur and Huberman, 2012), and spam detection (Yu, Asur and Huberman, 2012; Lin et al., 2013; Sun et al., 2013). There is some evidence that Sina Weibo users provide more information about themselves and are more episodic users than individuals on Twitter (Gao et al., 2012). Perhaps the most unique opportunity the platform affords academics is the ability to study censorship in real time and on a scale not before possible (Hassid, 2012; King, Pan and Roberts, 2014, 2016).

Some microblogging competitors to Twitter focus on image sharing. In its capabilities, Tumblr is equivalent to Twitter; in practice, Tumblr is used to share photos and GIFs (short,
low-quality videos), with accompanying text. The main difference is that Tumblr does not restrict the length of posts, whereas Twitter restricts to 140 characters; Twitter was designed with phones in mind, before smart phones existed, and it had to adhere to the character limit available to old phones’ text messages. Tumblr is also visually oriented, whereas Twitter has been dominated by text (though Twitter is becoming an image sharing platform as well). Though Tumblr has 555 million monthly active users, it has received less scholarly attention than Twitter (Social, 2016), perhaps because images are more difficult to analyze than text. For structural analysis of the platform, see Chang et al. (2014). Xu et al. show how the text component of Tumblr can help predict civil wars (Xu et al., 2014), and some work uses Tumblr for sentiment analysis (Bourlai and Herring, 2014). There appear to be no published works which analyze the images shared on Tumblr.

Launched in 2010 as a smart phone application, Instagram has become the most popular photo sharing platform, with over 400 million monthly active users. Users post and spread images they produce or find, and text is provided in the form of hashtags on each post. Academics have studied Instagram more than Tumblr but less than Facebook or Twitter. Images provide insight mass behaviors at specific times and places as well as document different cultural behaviors (Hochman and Manovich, 2013; Silva et al., 2013). For a structural overview of the Instagram network, see Ferrara et al. (2014). For an ontology of users and photographs on the site, see Hu et al. (2014). As of November 17, 2015, Instagram does not let third parties (including researchers) access its feed data, which is how you would download posts as they are created.

The best data on offline social networks comes from call detail records, the data telecommunications carriers record on who call and texts whom, the location of both parties, and the duration of the exchange. Because these data represent actual interactions, they should have more measurement validity than social networks recovered from online social media. A common critique of using social media to study social networks is that one has to assume that the social networks formed online have the same topography as those offline. Call detail
records therefore represent a major advance in the measurement of social networks at scale.

The researcher must obtain call detail records on an *ad hoc* basis. No company regularly makes them available, and companies aggregate at different levels of resolution (call or cell phone tower, for example). Orange, the French telecommunications company, has run two “Data 4 Development” competitions where they release call detail records, once from the Ivory Coast and once from Senegal. (They have not announced a regular schedule of competitions.) Finally, a country usually has multiple telecommunications providers, and a research project usually has access to only one company’s.

There exists little published work using call detail records. Nicholas Eubank has used seven months’ data from 9 million subscribers to measure ethnic fractionalization in Zambia (Eubank, 2016), and Fotini Christia, Leon Yao, Stephen Wittles, and Jure Leskovec have started to analyzed calls from Yemen during the Arab Spring (Christia et al., 2015). Patterns of calls may also predict violence (Berger, Kalyanaraman and Linardi, 2014). Outside of political science, patterns of calling are shown to track international trade but also reflect non-economic behaviors (Blumenstock, 2011), provide insight on international migration (Blumenstock, 2012), predict subnational poverty (Blumenstock, Cadamuro and On, 2015), and measure population distribution in real-time and in areas difficult to survey (Douglass et al., 2015).

6 Discussion

I conclude by discussing non-programmatic aspects of Twitter. The first subsection discusses the types of data that tweets do not provide and the limitations thus imposed on analysis. The second subsection raises potential ethical concerns of using Twitter data, especially as it relates to minors. I then user behaviors on Twitter to argue that it has features of both a media platform, like newspapers or television, and a social network.
6.1 Limitations

While the works above which are skeptical of Twitter enumerate shortcomings of the platform’s data, it is worth emphasizing them here as well. The simplest way to summarize these shortcomings is that individual tweets contain little information.

The main reason tweets have little information is because they are limited to 140 characters, 20 characters fewer than a text message. As any quick perusal of twitter.com reveals, this restriction leads to frequent use of abbreviations; it is also common for a tweet to be a comment on a link shared in the tweet. What tweets lack in information they make up for in quantity. One tweet may be about football, another music, and another a political candidate; individually, they are not interesting, but aggregated, they reveal interesting patterns about what topics are salient to a given group of people and how that saliency varies by place and time.

One of the drawbacks of Twitter for researchers is one of its appeals for users: anonymity. Registration is free, and registrants can make their screen name any word or phrase they want. Unlike Facebook, then, where you are asked for your first and last name, Twitter will let you appear to the world as “Zachary Steinert-Threlkeld” or “Brown Curtain”. While many users choose a screen name that is their name, most do not. Moreover, Twitter does not ask the users their age, education, gender, or race. To overcome this, researchers can take a sample of accounts and manually research them, generating reasonable estimates of accounts’ profession, age, gender, and political beliefs (Lotan et al., 2011; Borge-Holthoefer et al., 2015; Driscoll and Steinert-Threlkeld, 2017). Though much metadata is obtainable via research, such an approach does not scale to large populations.

Twitter users also have to decide to identify their location. They can identify in two ways. First, Twitter asks users their location as part of their profile. Like their user name, however, they can choose any word or phrase they want. While many users choose a real place, as many do not complete this section or give nonsensical answers such as ”the sky” or ”over there”. Second, Twitter users can assign GPS coordinates to their tweets, but they
have to choose to do this per tweet. 2%-3% of tweets have GPS coordinates attached to them (Leetaru et al., 2013; Budak and Watts, 2015). While a small percentage of tweets contain GPS coordinates, there are 500 million tweets per day. Users who tag their tweets with GPS coordinates do vary from those who do not, but that variation is only understood in the United States (Malik, Lamba, Nakos and Pfeffer, 2015), can be measured in many other countries, and does not necessarily mean the behavior in which the researcher is interested will be biased because the sample is.

To increase the number of geolocated tweets in a sample, scholars infer location based on a user’s self-reported location in their profile, though the user does not have to identify their location. This location can be any text string, so some users report their location as “Space” or “Under the Sea”, but most reported locations correspond to actual places. Including users’ self-reported location, anywhere from 34% (Leetaru et al., 2013) to 50% (Conover et al., 2013) to 66% (Hecht et al., 2011) of tweets have location information, the vast majority of which are at the city level. It is possible to assign location information to accounts based on the context of their tweets (Cheng, Caverlee and Lee, 2010; Hecht et al., 2011; Stefanidis, Crooks and Radzikowski, 2011). It is unknown whether accounts with self-reported location differ from those without, so the benefit of this added parsing needs to weighed against the computational cost. The most global source of location is an account’s time zone, as Twitter automatically assigns a time zone to each tweet based on the location where the tweet author registered (Lotan et al., 2011).

There are two idiosyncrasies to working with tweet text. First, Twitter requires all messages to contain less than 140 characters. The benefit of this limitation is that it tweets tend to concern themselves with one topic, simplifying analysis. The drawback of this limitation is that it pushes users to use abbreviations and slang, which complicate language processing. Second, tweet style is bimodal, with very many, perhaps a majority, of them using abbreviations and slang. Existing corpora used for dictionary approaches do not include slang, and the idiosyncratic nature of slang means unsupervised approaches are
more likely to assign tweets about the same topic to different collections. For a more detailed discussion on natural language processing and Twitter, see Sriram et. al. (2010) and Han and Baldwin (2011). For a thorough introduction to natural language processing more broadly, see Manning and Schütze (1999).

6.2 Ethics

Ethical concerns around the use of Twitter data flow from the scarcity of information in tweets and user profiles.

Because Twitter requires no identifying information to register, it is possible that tweets in one’s dataset are from children. Because Twitter is a common marketing and branding tool, many products exist which will estimate demographic age of an account’s followers. (Twitter also allows brands to require accounts which want to follow it to confirm their age before the potential follower is allowed to follow.) These products cost a lot of money. It is also unknown what percent of users on Twitter are under 18; Pew, which conducts an annual survey of Americans’ social media usage, does not interview minors. While there is a literature on estimating demographic characteristics from Twitter accounts’ behavior (Nguyen et al., 2013; Sloan et al., 2015; Sloan and Morgan, 2015), these solutions are not trivial to implement, and I am not aware of any academic Twitter studies which attempts to remove minors (or other protected categories) from their samples.

The researcher must also be careful to respect users’ desire to delete tweets. If a user deletes a specific tweet, the streaming API will deliver a JSON message with the tweet ID of the now deleted tweet. It is incumbent on the person or group connected to the API to delete the tweet from their data. Twitter is unclear about whether or not the streaming API provides the identification number of all deleted tweets or only a sample of them. Twitter will not make a tweet available via the REST API.

IRBs have not established a common standard for the treatment of Twitter data. When tweets are publicly available and researchers are not conducting interventions, there is no
prima facie reason studies should need IRB approval. Twitter’s public nature has not stopped IRBs from expressing caution about using its data (Halavais, 2011; Hayden, 2013). For a project where my colleague interviewed activists in Egypt and I examined their Twitter behavior, my university’s IRB had to approve the fieldwork (Fowler and Steinert-Threlkeld, 2016). When our IRB application mentioned Twitter, they asked for more detail, though Twitter’s public nature mollified them.

How IRBs approach minimally invasive research, such as downloading public data from Twitter, is in flux. On January 19, 2017, United States Government agencies in charge of protecting human subjects issued new guidelines for research that will take effect in January 2018. These guidelines create new exempt categories that require minimal IRB review, and research under these categories do not require continuing review. One of these categories is the “observation and recording of verbal and nonverbal behavior in schools and public places”, under which observational Twitter research should fall Shweder and Nisbett (2017). How institutions interpret these rules and the effect they have on the administrative cost of research remains to be seen.

While IRBs appear to have adopted an appropriate attitude to observational data from Twitter, there is growing interest in conducting experiments on the platform (Coppock, Guess and Ternovski, 2016; Munger, 2016). Procedures to protect subjects on Twitter appear to be the same as those for offline experiments. For example, the replication data for Munger (2016) is anonymized and aggregated to the account level so that the treated accounts cannot be identified. Coppock, Guess and Ternovski (2016) similarly do not share individual tweets, and personally identifiable information is removed from all replication material. Both papers’ authors received IRB approval before conducting their respective study.

Using tweets raises concerns about research reproducibility. A researcher cannot share more than 50,000 tweets per day. This restriction means that if the researcher has a dataset of 5 million tweets, it would take 100 days to transfer them to someone interested in reproduction. Relying on an individual to share tweets piecemeal does not encourage reproducible
research, as the researcher will have to manually slice his or her data and remember to regularly share those tweets with interested parties. Interested parties may be dissuaded from enquiring in the first place, however, since the opportunity and coordination costs are not trivial.

Moreover, shared tweets cannot be provided via “non-automated means”, and Twitter does not define that phrase. For example, suppose an academic publisher creates a script that automatically parses a tweet file into chunks of 50,000 tweets and tasks a staff member with sharing those chunks when requested. That script is an automated process, but the distribution mechanism (the staffer) is not. Now, suppose the publisher writes another script that e-mails those chunks to interested parties, but the staffer still has to initiate the chunking and e-mail scripts. Both solutions are less automated than allowing an interested party to initiate a series of downloads via a website interface, but the end result is the same. While it is clear that Twitter wants to impose costs on sharing data, these nonetheless exist an indeterminate range in which those costs could fall.

The permitted workaround is to share the identification numbers for tweets, as an infinite number of those can be shared. This approach is easy for the original researcher and is the current best practice, but it requires greater computational skill for the interested party. Scripts provided in this Element, however, download tweets by their identification number.

Finally, researchers should carefully consider how to handle tweets created by bots. While the word “bot” has a nefarious connotation after the 2016 United States Presidential Election, a bot is simply an account which behaves following prescribed rules. It is quite common, for example, to see bots that tweet the weather, financial news, or sports scores. Spam accounts tend to be bots, and the earliest attempt to screen for bots was to flag any tweet from an account less than one day old or that contains three trending topics Kwak et al. (2010). Scholars soon realized that bots could also be used for political purposes, and they have been widely by state and private political actors since at least 2011 Woolley (2016).

As Twitter has evolved, so have bots, and bot detection has become an active area of
research. One approach to identifying them is to use the source field in each tweet, as some sources are websites that allow customers to create and deploy bots (Forelle et al., 2015). Other approaches rely on community detection algorithms, crowdsourced manual labeling, or supervised machine learning Ferrara et al. (2016b). The machine learning approach finds that “social bots”, bots designed to behave like humans, retweet more than human accounts, have longer usernames, and are younger; they tweet, reply and mention others, and are retweeted less than human accounts Ferrara et al. (2016b). The model on which Ferrara et al. (2016b) rely to analyze their sample is available through Indiana University’s Observatory on Social Media’s “Bot or Not?” project (Ferrara et al., 2016a). Despite these advances in detection, bots continue to increase in sophistication and reach, and it is not clear that platforms will reach a point where they can eliminate bots. That bots require little financial capital makes their continued existence even more likely (Robertson, 2016; Bernstein, 2017).

While bots can be identified, their prevalence is an open question. During the 2016 United States Presidential Election, they were 14.4% of the top 50,000 users talking about Candidates Clinton and Trump and accounted for 18.45% of tweets about them Ferrara and Bessi (2016). Lotan et al. (2011) find 6% of accounts in Egypt in January 2011 were bots, 8.4% in Tunisia. In my experience, bots are much less prevalent when samples are restricted to accounts with GPS coordinates or profile locations. Of accounts tweeting from Ukraine during six months in 2014, they were a negligible percentage, and the Kwak et al. (2010) filter found only about 3% of tweets during the Arab Spring were spam Driscoll and Steinert-Threlkeld (2017); Steinert-Threlkeld (2017b). The more esoteric a research topic and the more prevalent geotagged tweets are in a sample, the less likely are bots to constitute a worrisome presence. Finally, once identified, it is preferable to model bots’ behavior and their effect on the larger social network than remove their tweets.
6.3 Is Twitter a Social Network or Media Platform?

Twitter is both a social network and a broadcast medium. It is certainly used by news organizations, politicians, celebrities, and corporations to broadcast messages, but it is also used by normal people engaging in activities that resemble how we think people behaved before they could document themselves on Twitter. Twitter, and probably social media in general, is both a broadcast medium and a social network.

One of the most famous studies of Twitter, and one of the first, directly addresses the site’s dual nature. Starting in June 2009, Haewoon Kwak and his coauthors crawled the site’s 41.7 million profiles, 1.47 billion social ties, and 106 million tweets to study information diffusion on the network (Kwak et al., 2010). Though their paper is titled, “What is Twitter, a Social Network or a News Media?”, they do not directly answer the question; they show, however, that it is both, depending on the characteristic analyzed.

Like a social network, the distribution of followers conforms to a power-law distribution. The distribution of followings, however, does not, with noticeable irregularities at 20 and 2000 followings (due to design features of Twitter). Very few accounts follow more than 1,000 accounts, and those that do tend to be service oriented ones such as politicians and companies. Over 85% of trending topics are headline or persistent news, consistent with a media platform. On the other hand, a retweet reaches the same average number of users regardless of the number of followers of the tweet’s author, consistent with a social network. 67.6% of users are not followed by any of the accounts they follow, suggesting that these people use Twitter more to gather information than to engage socially. Twitter has a short diameter (4.12 average links between each account and all other accounts), which the authors interpret as support for the broadcast side of Twitter. Like social networks, users exhibit homophily, in this case with respect to their number of followers and time-zone. Finally, ranking accounts by their number of followers, their PageRank, and the number of times they are retweeted shows that the top 20 accounts in each tend to be news organizations or celebrities.
Though news organizations and celebrities dominate in terms of followers and retweets, that does not mean they dominate on other dimensions. For example, Steinert-Threlkeld (2016) finds that those with the most followers did not drive protest mobilization during the Arab Spring. Even though those accounts will tweet about an upcoming or ongoing protest, the need for a critical mass of protestors means that it is the use of hashtags by those not at the top of the follower distribution that correlates with subsequent protest mobilization (Marwell, Oliver and Prahl, 1988). This result is in line with Barbera et al. (2015). They find that communication around the 2014 Academy Awards and raising the United States’ minimum wage resembles a broadcast network, while that for collective action has the same network dynamics identified in Steinert-Threlkeld (2016). Gonzalez-Bailon et al. (2013) finds four major types of Twitter users. Two of them - broadcasters (follow many fewer accounts than follow them, mentioned infrequently) and influentials (follow many fewer accounts than follow them, mentioned frequently) - are consistent with a media platform. The other two - common users (follow many more accounts than follow them, mentioned infrequently) and hidden influentials (follow many more accounts than follow them, mentioned frequently) - are consistent with a social network. Finally, a study of 1.8 billion tweets from four months in 2014 finds that only .8% of tweets are from news organizations, though some topics have up to 15% of their tweets coming from news organizations (Malik and Pfeffer, 2016). Individuals also commonly use Twitter to engage in public conversations (Honeycutt and Herring, 2009; Boyd, Golder and Lotan, 2010).

Intriguingly, interactions accounts have with each other on Twitter parallel how humans are known to behave offline. Dunbar’s Number is the observation that the number of social ties an animal can maintain is a function of the size of its neocortex in relation to the rest of the brain. For humans, this number is approximately 150 (Dunbar, 1992, 1995, 2011). Studying the reply and user mention behavior of 1.7 million networks, Bruno Goncalves et al. (2011) find that individuals on Twitter (bots were excluded) maintain between 100 and 200 connections (Gonçalves, Perra and Vespignani, 2011). The 150 offline contacts humans
maintain, however, exist in approximately three groups of increasing intimacy (Zhou et al., 2005). Three social media datasets, two from Facebook and one from Twitter, replicate this relationship hierarchy, though they find evidence of four to five layers (Dunbar et al., 2015).

Many papers’ findings would not make sense if Twitter was not used as a social network. For example, the ability to recover ideology scores based on follower relationships, and map those to voter records, would not be possible if users indiscriminately followed news organizations or political accounts (Barbera, 2015). Twitter users also exhibit less positive sentiment during working hours, consistent with social behaviors (Dodds et al., 2011; Golder and Macy, 2011), and express greater positivity when they tweet further from their most common locations (Frank et al., 2013). Exploiting the friendship paradox (your friends have more friends than you do because of the power-law distribution of friendship) allows one to detect emerging trends compared to monitoring a random sample of Twitter (Garcia-Herranz et al., 2014). Governments and individuals monitor Twitter during natural disasters to maintain situational awareness (Vieweg et al., 2010), and tweets even “detect” earthquakes (Sakaki, Okazaki and Matsuo, 2010). Twitter has also been used to predict crime in Chicago (Gerber, 2014) and map linguistic communities in New York City (Mocanu et al., 2013).

Since behaviors on Twitter resemble a broadcast platform and a social network, researchers should work to keep individual and broadcast accounts analytically distinct. There are two approaches to identifying broadcast accounts, and individual accounts are then those outside of that set. The first approach is to manually search Twitter for a specific broadcast account; most organizations, like the Los Angeles Times or NBC News, have Twitter accounts. The researcher should then feed the account names to the GET users/lookup endpoint of the REST API and record the user identification number. That identification number should be used to subsequent calls to the API, e.g. if one wants to follow that account from the streaming API. It is preferable to use the identification number instead of username because an account can change the latter but not the former. For example,
if the latimes changes to LosAngelesTimes, its identification number will not change. A researcher using the user name would have to update his or her code. Manually searching for accounts is time consuming, especially if one is many accounts or a category with which one is not familiar. The second approach takes advantage of Twitter lists to quicken the account identification. Twitter allows users to create publicly viewable lists of accounts, and users create lists around certain themes. For example, there are lists for French news and American cable television accounts. The best way to find these lists is via Google: in the search bar, type “site:twitter.com ‘search term’ list”, e.g. “site:twitter.com ‘France news’”. Once the list name is identified, the list members can be manually recorded or downloaded via the GET lists/members.

*****

For the foreseeable future, Twitter will continue to be the preferred data platform for social scientists, for two reasons. First, it remains a very large network. While it has not seen the growth envisioned at its initial public offering, its user base has held steady, as has its daily number of tweets. Controversies over its use for ISIS recruiting and the presence of racist and sexist harassers reflect real concerns, but those concerns have yet to affect the service’s popularity. Second, no other platform provides data as easily as Twitter. Instagram could have threatened Twitter for researchers’ loyalty, but its enclosure at the end of 2015 has taken away that prospect. Researchers interested in large datasets are now back to where they were in the mid-2000s: creating data by scraping blogs and news aggregators. Unless the researcher uses Twitter.

The strongest growth prospect for Twitter is live events coverage. While people already turn to it to consume breaking news, and governments and news organizations use it to disseminate news, Twitter is working to expand into live video broadcasts. In 2016, it started to broadcast Thursday night National Football League games, and it livestreamed

---

5Facebook and Instagram are great sources of data, if you can convince them to work with you and are willing to risk their veto power.
the 2016 United States Presidential debates (as did Facebook). It has also created a stand-
alone application, Periscope, that lets anyone broadcast live video from their phone. Because
Periscope broadcasts are shown via Twitter, one possibility is to see if the number of video
streams of certain events, like protests or riots, correlates with the actual number of attendees
of those events. Researchers in media consumption have had to rely on self-reported surveys
like Nielsen, so being able to observe in real time who observes broadcasts may present
exciting new research opportunities.

These data are not a “revolution”. Instead, they represent the next stage in the con-
stant increase in data available to researchers. In the 1970s, cutting-edge empirical research
consisted of descriptive statistics and basic regressions; using a computer required access
to a mainframe and programming with punch cards. The 1980s and 1990s democratized
computers for professionals, though data analysis programs did not become usable to those
without programming knowledge until the 1990s. The last two decades have witnessed the
rise of digital communications data. Because these data are much cheaper to produce than
human-created datasets, the amount of data available for analysis has expanded. This ex-
ansion has made previous approaches - aggregating data to the national or annual level,
modeling data through graphical user interfaces such as Stata or SPSS, or loading an en-
tire dataset at once, for example - often inadequate. Now, to stay at the forefront of data
analysis, one needs to know some programming in order to interface with websites and data
services, download data automatically, algorithmically clean and analyze data, and present
these data in low-dimension environments. The skills are modern; the change is eternal.
References


Analytics, Caerus. 2015. “Open Event Data Alliance.”.

URL: phoenixdata.org


URL: http://arxiv.org/abs/1611.09942


URL: https://cran.r-project.org/web/packages/streamR/

Barberá, Pablo. 2014. “How Social Media Reduces Mass Political Polarization. Evidence from Germany, Spain, and the U.S.”.


URL: http://www.ncbi.nlm.nih.gov/pubmed/26297377


Bergstrom, Kelly. 2011. “”Don’t feed the troll”: Shutting down debate about community expectations on Reddit.com.” *First Monday* 16(8).


URL: https://www.buzzfeed.com/josephbernstein/from-utah-with-love?utm_term=.xqpxB9kRv#.tiDymBqG7


Bird, Steven. 2015. “Natural Language Toolkit.”.


**URL:** http://dx.doi.org/10.7910/DVN/28075


**URL:** https://www.foreignaffairs.com/articles/yemen/2015-07-06/yemen-calling


**URL:** [http://dx.doi.org/10.1007/s11109-015-9308-6](http://dx.doi.org/10.1007/s11109-015-9308-6)


Diaz, Fernando, Michael Gamon, Jake Hofman, Emre Kiciman and David Rothschild. 2014. “Online and social media data as a flawed continuous panel survey.”.


Dowle, Matt, T Short, S Lianoglou and A Srinivasan. 2015. “data.table: Extension of data.frame.”.

**URL:** [https://cran.r-project.org/web/packages/data.table/index.html](https://cran.r-project.org/web/packages/data.table/index.html)

Driscoll, Jesse and Zachary C. Steinert-Threlkeld. 2017. “Structure, Agency, Hegemony, and Action: Ukrainian Nationalism in East Ukraine.”.


Duggan, Maeve, Dana Page and Senior Communications Manager. 2016. “Social Media Update 2016.”.


**URL:** [http://www.journals.cambridge.org/abstract_S1744137410000366](http://www.journals.cambridge.org/abstract_S1744137410000366)


Eubank, Nicholas. 2016. “Social Networks and the Political Salience of Ethnicity.”.


**URL:** http://www.iie.org/en/Research-and-Publications/Publications-and-Reports/IIE-Bookstore/DFG-UCSD-Publication#.V-MIM5MrKqA


Freelon, Dean. 2012. “Arab Spring Twitter data now available (sort of).”.

**URL:** http://dfreonl.org/2012/02/11/arab-spring-twitter-data-now-available-sort-of/


Gentry, Jeff. 2015. “twitteR.”. 
**URL:** [https://cran.r-project.org/web/packages/twitteR/index.html](https://cran.r-project.org/web/packages/twitteR/index.html)


González-Bailón, Sandra, Ning Wang, Alejandro Rivero, Javier Borge-Holthoefer and Yamir Moreno. 2012. “Assessing the Bias in Communication Networks Sampled from Twitter.”.


Groshek, Jacob. 2015. “Status Update on the BU-TCAT.”. 

Halavais, Alexander. 2011. “Social science: open up online research.”


Hayden, Erika Check. 2013. “Guidance issued for US internet research: Institutional review boards may need to takea closer look at some types of online research.”

URL: [http://www.nature.com/news/guidance-issued-for-us-internet-research-1.12860](http://www.nature.com/news/guidance-issued-for-us-internet-research-1.12860)

Hecht, Brent, Lichan Hong, Bongwon Suh and Ed H. Chi. 2011. Tweets from Justin Bieber’s Heart: The Dynamics of the Location Field in User Profiles. In *ACM Conference on Human Factors in Computing Systems*. Number Figure 1 Vancouver:


Lakkaraju, Himabindu, Julian J. McAuley and Jure Leskovec. 2013. What’s in a Name? Understanding the Interplay between Titles, Content, and Communities in Social Media. In ICWSM.

Lang, Duncam Temple. 2015. “RCurl.”.


URL: http://linkinghub.elsevier.com/retrieve/pii/S0378873308000385


Llorente, Alejandro, Manuel Garcia-Herranz, Manuel Cebrian and Esteban Moro. 2014. “Social media fingerprints of unemployment.”.
URL: http://arxiv.org/abs/1411.3140


Masad, David. 2013. “Studying the Syrian Civil War with GDELT.”.


McGrath, Ryan. 2015. “twython.”.

McKinney, Wes. 2015. “pandas.”.
URL: http://pandas.pydata.org/


Nguyen, Dong, Rilana Gravel, Dolf Trieschnigg and Theo Meder. 2013. “”How old do you think I am ?: A study of language and age in Twitter.”.


Seabold, Skipper and Josep Perktold. 2014. “statstools.”. URL: https://pypi.python.org/pypi/statsmodels


Steinert-Threlkeld, Zachary C. 2016. “Replication Data for: Longitudinal Network Centrality Using Incomplete Data.”. URL: http://dx.doi.org/10.7910/DVN/KKWB4A

Steinert-Threlkeld, Zachary C. 2017a. “Longitudinal Network Analysis with Incomplete Data.” Political Analysis.

Steinert-Threlkeld, Zachary C. 2017b. “Spontaneous Collective Action: Peripheral Mobilization During the Arab Spring.” American Political Science Review 111(02).


URL: http://www.pewinternet.org/2013/09/12/location-based-services/