Analyzing tweets to detect social media events

Frederick J. Boehm and Bret M. Hanlon

June 15, 2020

Introduction

Twitter has profoundly changed how we communicate. In only 280 characters, users can instantly contribute to public conversations on politics, current events, sports, media, and many other topics. Recent development of accessible statistical methods for large-scale text analysis now enable instructors to use tweets as contemporary pedagogical tools in guiding undergraduate research projects. We report one instance of a mentored text analysis research project. We share our data and computer code to encourage others to undertake tweet text analysis research. We also describe our methods for creating a collection of tweets.

Some social media data, including tweets from Twitter, is available through website application product interfaces (APIs). By way of a streaming API, Twitter shares a sample of approximately one percent of all tweets during an API query time period.[1] Any Twitter user can freely access this one percent sample, whereas access to a larger selection is available to researchers for a fee.

Using large collections of tweets, scholars have studied diverse research questions, including the inference of relationships and social networks among Twitter users;[2] authorship of specific tweets when multiple persons share a single account;[3] and rhetoric in recruiting political supporters.[4] Recognizing the potential utility of tweets for data science research and teaching, we created a collection of tweets over time by repeated querying of the Twitter streaming API.

In line with recent calls for students to work with real data,[5] our collection of tweets has served as a valuable resource in our mentoring of undergraduate data science research. Working with real data allows students to develop proficiency not only in statistical analysis, but also in related data science skills, including data

[1] "Sampled Stream."
[2] Lin et al., "The Joint Inference of Topic Diffusion and Evolution in Social Communities."
[3] Robinson, "Text Analysis of Trump's Tweets Confirms He Writes Only the (Angrier) Android Half."
[4] Pelled et al., "'Little Marco,''Lyin'Ted,''Crooked Hillary,' and the 'Biased' Media"; Wells et al., "How Trump Drove Coverage to the Nomination."
[5] Nolan and Temple Lang, "Computing in the Statistics Curricula."

transfer from online sources, data storage, using data from multiple file formats, and communicating findings and their limitations. Collaboratively asking and addressing novel questions with our collection of tweets gave mentored students opportunities to develop competency in all of these areas.

While our tweet collection enables us to address many possible research questions, the dynamic content of tweets over time particularly piqued our interest. Together, students and mentors hypothesized that high-profile social media events would generate a high volume of tweets, and that we would detect social media events through changes in tweet topic content over time. We present below 1) an approach for collecting tweets in real time and 2) statistical methods for detecting social media events via latent Dirichlet allocation modeling of collections of tweets and 3) reflections on using this data set in research mentoring of undergraduate students.

Research mentoring considerations

We collaboratively developed research goals with students through a series of discussions throughout the academic year. As trainees begin their senior research projects, we suggest that mentors discuss with them:

Student research interests and goals

Student experience with data analysis and statistical analysis software Software for statistical analysis and best practices for reproducibility

Student research interests and goals

In our experience, student interests vary, and their initial ability to articulate research goals may be limited. An initial brainstorming session may clarify their interests and encourage them to think critically about goals under the time constraints of their academic schedules. Additionally, we anticipate that sharing completed student project reports will guide student thinking about the scope of possible projects.

We mentored one student whose interest in financial time series guided her project. A second student formulated a project around event detection from tweet time series.

Student experience with data analysis and statistical software

Software for data

We used the linux software package `crontab` to automate the execution of R code every five minutes. The tweets are available from the Twitter API as JSON. We saved the JSON in plain text files.

We performed all analyses in the R statistical computing environment. We used the Rstudio software package to interface with R. We share our R code to enhance reproducibility of our findings. Users may also find our code helpful when creating their own tweet collections.

Backward design

We sought to give our trainees opportunities to formulate and address a data science research question. We consulted the GAISE report for additional guidance. The GAISE report executive summary highlights six goals for statistics training:

1. Teach statistical thinking

2. Focus on conceptual understanding

3. Integrate real data with a context and purpose

4. Foster active learning

5. Use technology to explore concepts and analyze data

6. Use assessments to improve and evaluate student learning

Learning outcomes

Time period

We began collecting tweets in 2013. Our two students conducted their research projects during the 2015-2016 academic year. We recommend a full academic year for projects of this magnitude, although a one-semester project is possible. Our students presented their findings at the statistics department's undergraduate poster session near the end of the 2015-2016 academic year.

Methods

Collecting tweets over time

We include here instructions for creating a tweet collection. First, we created a new account on Twitter. With these user credentials, we used the R package `rtweet` to query the API. Because we work with linux operating systems, we used the `crontab` software to repeatedly execute R code to submit API queries. Each

query lasted five minutes. We timed the API queries so that there was no time lag between queries. We stored tweets resulting from API queries in their native JSON format.

The R package `rtweet` provides functions that parse tweet JSON to R data frames. We then conducted all further analyses in R.

Setting up the query task with `crontab` is straightforward. On our computer, with Ubuntu 20.04 linux operating system, we opened a terminal and typed `crontab -e`. This opened a text file containing user-specified tasks. We added the following line to the bottom of the file:

```
*/5 * * * * R -e 'rtweet::stream_tweets(timeout = (60 * 5),
parse = FALSE, file_name = paste0("~/work/mentoring/mentoring-framework/data/",
lubridate::now(), "-tweets"))'
```

Users may need to slightly amend the above line to conform to requirements of their operating system's `crontab`.

Querying Twitter API to get complete tweets

Twitter API use agreements forbid users from sharing complete API query results. However, Twitter enables users to share tweet identification numbers. A user can then query a Twitter API to obtain complete tweet data. In our experience, this process is incomplete; that is, many tweets submitted to the Twitter API return no data. Additionally, on repeated querying of the API, different sets of tweets return data. This complicates our goal of making all analyses computationally reproducible.

From our collection of tweets, we chose to analyze those sent on five consecutive days from May 10, 2020 to May 14, 2020. To minimize the computing requirements, we limited our study to tweets sent during a five-minute period (12:00pm to 12:05pm Eastern time) every day. We then submitted API queries to Twitter to get the full content of tweets, including the tweet text. In supplementary files, we provide the R code that we used to query the Twitter API to obtain full tweet content.

Tweet structure

Tweets are available from the Twitter API as Javascript Object Notation (JSON) objects. Every tweet consists of multiple key-value pairs. The number of fields per tweet depends on user settings, retweet status, and other factors.[6] The 31 tweet key-value pairs belong to 12 distinct classes (Appendix 1). The classes are either vectors - numeric, logical, or character - or arrays assembled from the vector classes.

[6] "Introduction to Tweet Json."

Below is an example of Tweet JSON. Every tweet features the keys "created_at" (the time stamp), "id_str" (a unique tweet identifier), and "text". We use these three keys in our analyses.


```
{
  "created_at": "Thu Apr 06 15:24:15 +0000 2017",
  "id_str": "850006245121695744",
  "text": "1\/ Today we\u2019re sharing our vision for the future of the Twitter API platform!\nhttps:\/
  "user": {
    "id": 2244994945,
    "name": "Twitter Dev",
    "screen_name": "TwitterDev",
    "location": "Internet",
    "url": "https:\/\/dev.twitter.com\/",
    "description": "Your official source for Twitter Platform news, updates & events.
    Need technical help? Visit https:\/\/twittercommunity.com\/ \u2328\ufe0f
    #TapIntoTwitter"
  },
  "place": {
  },
  "entities": {
    "hashtags": [
    ],
    "urls": [
      {
        "url": "https:\/\/t.co\/XweGngmxlP",
        "unwound": {
          "url": "https:\/\/cards.twitter.com\/cards\/18ce53wgo4h\/3xo1c",
          "title": "Building the Future of the Twitter API Platform"
        }
      }
    ],
    "user_mentions": [
    ]
```

```
  }
}
```

Parsing text of tweets

We used functions from the `rtweet` R package to parse tweet JSON into a data frame. With `rtweet` functions, we parsed JSON arrays into their component vectors and added them to the data frame.

We then divided tweet text into words with functions from the `tidytext` R package. We discarded commonly used "stop words" and emojis.

Latent Dirichlet allocation models require that the corpus be inputted as a document by term matrix. Each row corresponds to a single document (a single tweet), and each column is a single term (or word). Each cell contains a count (the number of occurrences of a term in the specified document). We created a document by term matrix with the R function `cast_dtm` from the `tidytext` package.

Latent Dirichlet allocation

Latent Dirichlet allocation is a statistical method for inferring latent (unobservable) topics (or themes) from a corpus (or collection) of documents. We pretend that there's an imaginary process for creating documents in the corpus. For each document, we choose a discrete distribution over topics. For example, some Mother's Day tweets wish mothers a happy celebration. This may constitute one topic in the corpus. Having chosen a distribution over topics, we then select document words by first drawing a topic from the distribution over topics, then drawing a word from the chosen topic. Thus, a topic is technically defined as a distribution over words in a fixed vocabulary (or collection of words).

The literature on latent Dirichlet allocation and related methods is vast, and we won't attempt to review it here.

Study design

We sought to validate our hypothesis that we could detect a major social media event by examining tweet topic content at distinct time periods. As a proof of principle of our event detection strategy, we chose to analyze tweets during and after Mother's Day 2020. We fitted latent Dirichlet allocation models for each of four distinct five-minute periods. The first period began at noon Eastern time on Mother's Day 2020. Subsequent time periods started 24, 48, and 72 hours later. We defined each time period to be a single collection, or corpus, of tweets. We then fitted latent Dirichlet allocation models to each corpus.

We used several criteria to evaluate latent Dirichlet allocation model fits, with emphasis on choosing a reasonable number of topics per corpus. Our strategy involved both visualization and more quantitative approaches to model evaluation. For every model, we created one word cloud per topic.

We then inspected topic contents at each of the four time points.

References

"Introduction to Tweet Json." https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json, 2020
Accessed: 2020-05-18.

Lin, Cindy Xide, Qiaozhu Mei, Jiawei Han, Yunliang Jiang, and Marina Danilevsky. "The Joint Inference of Topic Diffusion and Evolution in Social Communities." In *2011 Ieee 11th International Conference on Data Mining*, 378–87. IEEE, 2011.

Nolan, Deborah, and Duncan Temple Lang. "Computing in the Statistics Curricula." *The American Statistician* 64, no. 2 (2010): 97–107.

Pelled, Ayellet, Josephine Lukito, Fred Boehm, JungHwan Yang, and Dhavan Shah. "'Little Marco,''Lyin'Ted,''Crooked Hillary,' and the 'Biased' Media: How Trump Used Twitter to Attack and Organize." In *Digital Discussions*, 176–96. Routledge, 2018.

Robinson, David. "Text Analysis of Trump's Tweets Confirms He Writes Only the (Angrier) Android Half." http://varianceexplained.org/r/trump-tweets/, 2016
Accessed: 2019-11-26.

"Sampled Stream." https://developer.twitter.com/en/docs/labs/sampled-stream/overview, 2019
Accessed: 2019-11-27.

Wells, Chris, Dhavan V Shah, Jon C Pevehouse, JungHwan Yang, Ayellet Pelled, Frederick Boehm, Josephine Lukito, Shreenita Ghosh, and Jessica L Schmidt. "How Trump Drove Coverage to the Nomination: Hybrid Media Campaigning." *Political Communication* 33, no. 4 (2016): 669–76.

Appendix 1: Tweet data dictionary

Twitter shares a data dictionary for tweets (https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/tweet-object, (Accessed: May 23, 2020)). We have saved it as a supplementary file, "tweets-data-dictionary.csv".