README

# Overview

The ***MITLL TweetE Twitter Analysis Tools*** perform multiple types of analysis on Twitter data:

* Unstructured tweets to structured data and text normalization
* Twitter graph creation

Tweets are ingested from a flat TSV (tab-separated value) file. Results are stored in a serialized Python object (text analysis and normalization) and multiple graph formats. Examples of research applications that used these tools are contained in the papers [WCampbell13] and [WCampbell14].

Provided a collection of tweets, the *MITLL TweetE Twitter Analysis Tools*:

1. Normalize the input text and remove links and non-language characters
2. Extract information: hashtags, links, at-mentions
3. Filter out all documents not matching the user-specified language
4. Filter by geo location
5. Store the results in serialized files for graph creation, analysis with the MIT topic tools, or other counts-based classifiers
6. Create rich Twitter graphs

These tools are command-line applications mainly suited for researchers who would like to convert Twitter data into structured form for further high-level analysis—e.g., natural language processing and graph analysis.

# Download

The code for the MITLL Twitter Analysis Tools is publicly accessible through a git repository at <https://github.com/mitll/TweetE> . For help on *github* and *git*, please refer to <https://github.com> and <http://git-scm.com/> respectively.

Once you have downloaded the code, you can run the program by installing the dependencies listed in section ***Dependencies***, and following the steps described in section ***Running the <>***.

# Dependencies

The system is a command-line application for 64-bit Linux written in Python and C++. It requires the following dependencies:

* Python 2.7.x
* NumPy (Python module)
* NetworkX (<https://networkx.github.io/>). Install NetworkX for your local python version using the instructions at the NetworkX website.

# Target Platform

* 64-bit Linux

# Running the Twitter Ingester

The shell script

run\_ingest\_all.sh

shows a simple example of running the ingest tool.

In the scripts, the ingest tool takes “tsv.gz” files from the directory twitter/user\_tweets and converts them into serialized files in the twitter/serialized directory.

# Data Format and Ingestion

Twitter data is contained in tab-delimited UTF-8 text files that are gzipped. The ingester assumes the input file has one line per document in the following format:

<tweetid>TAB<date>TAB<language>TAB<geo-coordinates>TAB<username>TAB<tweet text>NEW\_LINE

A more detailed description of these fields can be found at the Twitter API website, <https://dev.twitter.com/overview/api/tweets> .

An example from the file aaastudio.tweets.tsv.gz is shown as follows:

286163982614679552 Tue Jan 01 17:36:24 +0000 2013 en (0.0, 0.0) aaastudio Happy New Year!! Wishing all a healthy, happy and prosperous 2013!!

286173401771565056 Tue Jan 01 18:13:50 +0000 2013 en (0.0, 0.0) aaastudio @RW2Photo fantastic

286338113477939200 Wed Jan 02 05:08:20 +0000 2013 und (0.0, 0.0) aaastudio Day 1 http://t.co/8W9h6tJZ

286898889729142784 Thu Jan 03 18:16:40 +0000 2013 es (0.0, 0.0) aaastudio Carlos (c)2012 Angel Alvarado / AAA Studio http://t.co/iOM5gH9Z

287696368833818624 Sat Jan 05 23:05:34 +0000 2013 en (0.0, 0.0) aaastudio In the middle of a shoot

# Output

After running the script ./run\_ingest\_all.sh in the twitter\_analysis directory, serialized files are saved to the twitter/serialized directory. The output can be displayed using the tool scripts/display\_tweets.py. An example is shown as follows:

$ scripts/display\_tweets.py --in twitter/serialized/tw\_user\_tweets\_aaastudio.pckl --output tmp/a.txt  
Reading in file: twitter/serialized/tw\_user\_tweets\_aaastudio.pckl  
Done

% head -18 tmp/a.txt

date Tue Aug 27 07:56:29 +0000 2013  
geo (0.0, 0.0)  
http\_links [(117,http://t.co/ao8Uj2DteD)]  
id 372266352943198208  
lid\_gnip en  
msg Really great news I've just upgraded to Spotify Premium I've got unlimited, ad-free music on my mobile and computer. <http://t.co/ao8Uj2DteD>  
msg\_norm Really great news I've just upgraded to Spotify Premium I've got unlimited ad-free music on my mobile and computer.  
userid aaastudio  
  
date Sun May 26 20:23:26 +0000 2013  
geo (40.76511645, -73.97349006)  
http\_links [(54,http://t.co/6X9OUPx3C7)]  
id 338752255736045569  
lid\_gnip en  
msg I'm at Central Park South (New York, NY) w/ 13 others <http://t.co/6X9OUPx3C7>  
msg\_norm I'm at Central Park South New York NY w 13 others  
userid aaastudio

From the example, we see that the system has extracted http\_links and has normalized the text. Further inspection of the structured output will show extracted at-mentions and hashtags.

# Running the Graph Creation Tool

The shell script

run\_graph\_all.sh

shows an advanced example of running the graph creation tool. The run\_ingest\_all.sh tool must be run before graph creation.

The script run\_graph\_all.sh is designed for processing a large amount of Twitter data and creating graphs in multiple steps. The steps are:

1. Take multiple serialized files and create a graph per list (*run\_p1* in the code)
2. Merge multiple graphs to create another set of graphs (*run\_p2* in the code)
3. Merge multiple graphs to create a node set and edge set with duplicates (*run\_p3* in the code)
4. Merge edge set duplicates (*run\_p4*)
5. Create various versions of the final graph by pruning nodes by weighted degree

A typical run might create 100 graphs in step 1, 10 graphs in step 2, 1 graph in step 3, perform an edge merger in step 4, and then prune with different node weights. If only analysis of a small amount of Twitter data is needed, then using the code in steps 1-2 may be sufficient (the original code will need to be modified).

# Output

After running the graph creation tool, the results will be stored in the directory graph/. For the example files the output files should be:

node\_degree.txt.gz twitter\_prune\_w10.edges.txt.gz

twitter\_prune\_w5.edges.txt.gz twitter\_all.edges.txt.gz twitter\_prune\_w10.nodes.txt.gz twitter\_prune\_w5.nodes.txt.gz twitter\_all.nodes.txt.gz twitter\_prune\_w1.edges.txt.gz

twitter\_prune\_w100.edges.txt.gz twitter\_prune\_w1.nodes.txt.gz twitter\_prune\_w50.edges.txt.gz

twitter\_prune\_w100.nodes.txt.gz twitter\_prune\_w50.nodes.txt.gz

The files twitter\_all\_\*.txt.gz contain the unpruned graphs and the files with twitter\_prune\_w<weight>.\*.txt.gz contain the versions pruned by node degree. An example few nodes in the node file are:

$ zcat graph/twitter\_all.nodes.txt.gz | head

1118 #truth  
12102 @jayray809  
7946 @hellocupkake  
6617 @tomcrabtree  
7947 @oglethewriter  
1119 #eminem  
1120 #dailyshow  
9733 @doingitwrong  
3827 @tomorrowsprjct  
11040 @spacekisser

The format of this file is <nodeid> <twitter hashtag/username> in space delimited format. Note that the node id is *not* the Twitter user id and is arbitrary. An example from the edge file is as shown:

zcat graph/twitter\_all.edges.txt.gz | head

0 6 1  
0 9 0 0 1  
0 10 6 0 3  
0 15 2 0 0  
0 20 2 0 1  
0 21 0 0 2  
0 24 4  
0 37 7  
0 39 2 0 0  
0 40 1

The format is <src node> <dest node> <w1> <w2> <w3>. The source and destination nodes refer to the node ids in the twitter\_all.nodes.txt.gz file. Note that the graph is directed and by design the adjacency matrix is not symmetric. The weights depend on the edge type. There are three cases:

* Hashtag -> Hashtag: w1=co-occurrence count of these hashtags
* User->Hashtag: w1=the number of times the hashtag occurred in tweets from this user
* User->User: w1=co-occurrence count of user names, w2=communication count of user src with user dest, w3=retweet of user dest by user src

More details on the node and edge structure and sample graph analysis can be found in [WCampbell13] and [WCampbell14].

# References

[WCampbell13] W. M. Campbell, E. Baseman, K. Greenfield, "Content+Context Networks for User Classification in Twitter," NIPS 2014 Workshop, Frontiers of Network Analysis: Methods, Models, and Applications, 2013.

[WCampbell14] W. M. Campbell, E. Baseman, K. Greenfield, "Content+Context=Classification: Examining the Roles of Social Interactions and Linguistic Content in Twitter User Classification," Coling 2014 Workshop on Natural Language Processing for Social Media (SocialNLP), 2014.