Web Science cs532: Assignment #6

Due on Thursday, March 17, 2016

Dr.Michael.L.Nelson 4:20pm

Zetan Li

Contents

Problem 1

Use D3 to visualize your Twitter followers. Use my twitter account ("@phonedude_mln") if you do not have >= 50 followers. For example, @hvdsomp follows me, as does @mart1nkle1n. They also follow each other, so they would both have links to me and links to each other.

To see if two users follow each other, see:
`https://dev.twitter.com/rest/reference/get/friendships/show`

Attractiveness of the graph counts! Nodes should be labeled (avatar images are even better), and edge types (follows, following) should be marked.

Note: for getting GitHub to serve HTML (and other media types), see:
`http://stackoverflow.com/questions/6551446/can-i-run-html-files-directly-from-github-inste`

Be sure to include the URI(s) for your D3 graph in your report.

SOLUTION

First, we have to get the followship data from twitter.
Tweepy have a function called api.show_friendship which wraps the REST API for followship query.

However, this query only allows a pair at a time, and twitter has the limit of 180 search terms per 18 minutes, the total query time is the combination of 2 out of total number of followers. So we have to minimize out data scale.

Here, we pick the follower "joc7188" as our test case. He's user name is Jose Antonio Olvera and he has 52 followers since we inspect his account.
To get the network graph of his follower, first we have to get all his follower accounts.
Below is the code to extract all his followers.

Listing 1: Python script to get all follower accounts

```
import tweepy
from tweepy import *
import time
import os
CONSUMER_KEY = "YwekEH9UrlXUFKUH2XmImE681"
CONSUMER_SECRET = "NWaB0jN5HaQ3f9VqCrMhP2nP8154KGXoPeDxZk6TIOVptAxErb"

OAUTH_TOKEN = "4860544225-qbjIQvrGlrj493eIHAjNu2OH0rdrHlM94XMHe1x"
OAUTH_TOKEN_SECRET = "vfAc4sJwbWExBjrMZiMqRMuknTzbl2le55DKX5gGYOOXR"

auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN,OAUTH_TOKEN_SECRET)
api=tweepy.API(auth)

fdata=open('followers.txt','w')
user=api.get_user('joc7188')
fdata.write('joc7188'+'\n')

for user in tweepy.Cursor(api.followers, screen_name="joc7188",count=200).items():
```

Problem 1 continued on next page...

```
     #users.append(user)
     fdata.write(user.screen_name+'\n')
fdata.close()
```

Now we have the list of all his followers. Next step is to inspect the followship between them.

api.show_friendship returns a tuple contains two friendship objects, which represent state of friendship of two account we queried. We can get "following" and "followed_by" from either of them.

Due to the rate limit of the twitter, this takes a long time to get all the data.

Listing 2: Python script to get followship between followers

```
import tweepy
from tweepy import *
import time
import os
CONSUMER_KEY = "YwekEH9UrlXUFKUH2XmImE681"
CONSUMER_SECRET = "NWaB0jN5HaQ3f9VqCrMhP2nP8154KGXoPeDxZk6TIOVptAxErb"

OAUTH_TOKEN = "4860544225-qbjIQvrGlrj493eIHAjNu2OH0rdrHlM94XMHe1x"
OAUTH_TOKEN_SECRET = "vfAc4sJwbWExBjrMZiMqRMuknTzbl2le55DKX5gGYOOXR"

auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN,OAUTH_TOKEN_SECRET)
api=tweepy.API(auth,wait_on_rate_limit =True, wait_on_rate_limit_notify=True)

fdata=open('followers.txt')
strlines=fdata.readlines()
usernames=list(line for line in strlines)
fdata.close()

outdata=open('net.txt','a')
for i in range(len(usernames)-1):
     for j in range(i+1,len(usernames)):
          while True:
               try:
                    fs=api.show_friendship(source_screen_name = usernames[i],
                        target_screen_name=usernames[j])
                    index1=i
                    index2=j
                    sname1=fs[0].screen_name
                    sname2=fs[1].screen_name
                    linktype=3
                    if fs[0].following or fs[0].followed_by :
                         if fs[0].following :
                              if fs[0].followed_by:
                                   linktype=2
                              else:
                                   linktype=0
                         else:
                              if fs[0].followed_by:
```

```
                                        linktype=1
                            if linktype!=3:
                                outdata.write(str(i)+'\t'+str(j)+'\t'+sname1+'\t'+
                                    sname2+'\t'+str(linktype)+'\n')
                        break
                        #0: -->; 1: <--; 2: <-->
                    except BaseException as e:
                        print (str(e))
                        if 'Not authorized' in str(e):
                            break
                        else:
                            time.sleep(5)
                            continue



outdata.close()
# fs=api.show_friendship(source_screen_name='joc7188',target_screen_name='
    ItsMyBell')
# print(fs[0])
# print(fs[1].following)
```

Right now, we have all the data we need to plot the d3 graph.
We could use d3.tsv to parse the file which separated by tab.
To build the links, we have to iterate each link in our data file, add link according to the link type. (In our data file, we defined 0 as following, 1 as followed by, and 2 as bidirectional following)
Then add each node to the graph and plot a curve [1] on each link (special thanks to rcond for his code inspired me on how to plot the markers).

Listing 3: Core code to construct the d3 graph

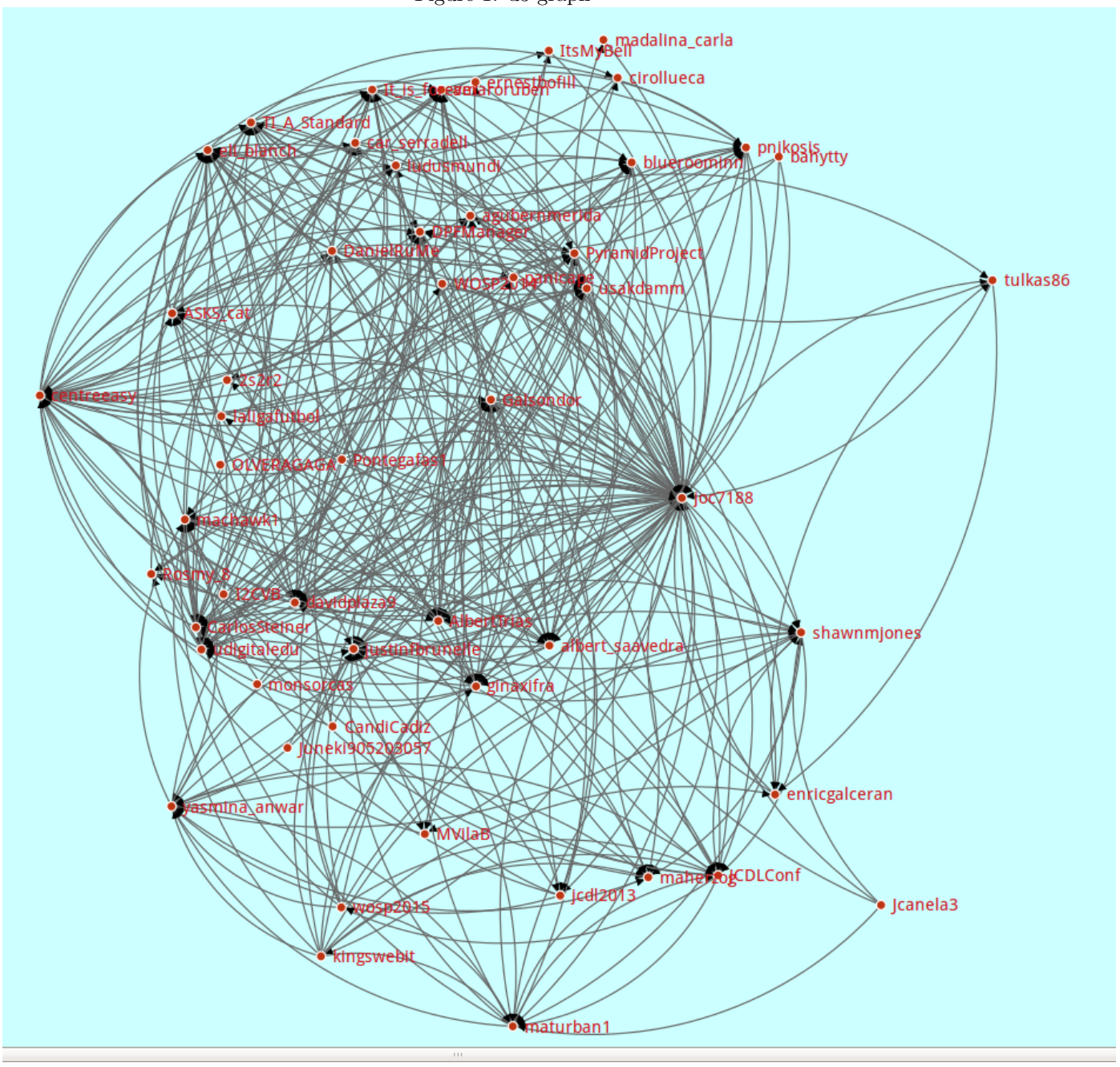
```
var width = 2000,
    height = 1500;

var color = d3.scale.category20();

var force = d3.layout.force()
    .charge(-350)
    .linkDistance(500)
    .size([width, height]);

var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height);

d3.tsv("net.tsv", function(error, graph) {
  if (error) throw error;

var nodes ={}
var links=[]
graph.forEach(
  function(link){
```

```
    link.source_name = nodes[link.source_name] ||
        (nodes[link.source_name] = {name: link.source_name,});
    link.target_name = nodes[link.target_name] ||
        (nodes[link.target_name] = {name: link.target_name});

        switch(link.link_type)
        {
          case '0':
          links.push({source: link.source_name, target: link.target_name });
          break;
          case '1':
          links.push({source: link.target_name , target: link.source_name});
          break;
          case '2':
          links.push({source: link.source_name, target: link.target_name });
          links.push({source: link.target_name , target: link.source_name});
          break;
          default:
          break;
        }
  }
);

  force
      .nodes(d3.values(nodes))
      .links(links)
      .start();

// build the arrow.
svg.append("svg:defs").selectAll("marker")
    .data(["end"])
  .enter().append("svg:marker")
    .attr("id", String)
    .attr("viewBox", "0 -5 10 10")
    .attr("refX", 15)
    .attr("refY", -1.5)
    .attr("markerWidth", 6)
    .attr("markerHeight", 6)
    .attr("orient", "auto")
  .append("svg:path")
    .attr("d", "M0,-5L10,0L0,5");

// add the links and the arrows
var path = svg.append("svg:g").selectAll("path")
    .data(force.links())
  .enter().append("svg:path")
    .attr("class", "link")
    .attr("marker-end", "url(#end)");

  var node = svg.selectAll(".node")
      .data(force.nodes())
    .enter().append("g")
      .attr("class", "node")
```

```
      .call(force.drag);
  // add the nodes
node.append("circle")
    .attr("r", 5);

  node.append("text")
    .attr("x", 12)
    .attr("dy", ".35em")
    .text(function(d) { return d.name; });

  force.on("tick", function() {
    path.attr("d", function(d) {
        var dx = d.target.x - d.source.x,
            dy = d.target.y - d.source.y,
            dr = Math.sqrt(dx * dx + dy * dy);
        return "M" +
            d.source.x + "," +
            d.source.y + "A" +
            dr + "," + dr + " 0 0,1 " +
            d.target.x + "," +
            d.target.y; });

    node
        .attr("transform", function(d) {
            return "translate(" + d.x + "," + d.y + ")"; });
  });
});
```

Below is the d3 graph screen capture, for online rendering, see `https://rawgit.com/DarkAngelZT/cs532-s16/master/assignments/a6/d3/twitter.html`

Figure 1: d3 graph



Problem 2

Take the Twitter graph you generated in question #1 and test for male-female homophily. For the purposes of this question you can consider the graph as undirected (i.e., no distinction between "follows" and "following"). Use the twitter name (not "screen name"; for example "Michael L. Nelson" and not "@phonedude_mln") and programatically determine if the user is male or female. Some sites that might be useful:

https://genderize.io/
https://pypi.python.org/pypi/gender-detector/0.0.4

Create a table of Twitter users and their likely gender. List any accounts that can't be determined and remove them from the graph.

Perform the homophily test as described in slides 11-15, Week 7.

Does your Twitter graph exhibit gender homophily?

SOLUTION

Since we have all the follower's account, It's easy to get their possible gender.
So far in our dataset, only the screen name is recorded. We have to get their real name first. Using api.get_user, then we can retrieve user name in the field called"user.name".

Listing 4: Python script to get user names

```
import tweepy
from tweepy import *
import time
import os
import sys
reload(sys)
sys.setdefaultencoding('utf-8')
CONSUMER_KEY = "YwekEH9UrlXUFKUH2XmImE681"
CONSUMER_SECRET = "NWaB0jN5HaQ3f9VqCrMhP2nP8154KGXoPeDxZk6TIOVptAxErb"

OAUTH_TOKEN = "4860544225-qbjIQvrGlrj493eIHAjNu2OH0rdrHlM94XMHe1x"
OAUTH_TOKEN_SECRET = "vfAc4sJwbWExBjrMZiMqRMuknTzbl2le55DKX5gGYOOXR"

auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN,OAUTH_TOKEN_SECRET)
api=tweepy.API(auth)

fdata=open('followers_name.txt','w')
user=api.get_user('joc7188')

fdata.write(user.name+'\n')

for user in tweepy.Cursor(api.followers, screen_name="joc7188",count=200).items():
    #users.append(user)
    fdata.write(user.name+'\n')
fdata.close()
```

Then we can use online service genderize.io to get their possible gender.[2]
Algorithm is like this:

1) Pair each account name and user name then get their gender via genderize.io, store all of them into data file.
2) Look up every link in the followship data table.
3) Extract two account name from each followship.
4) Look up their user name and get the gender from data file.
5) If two genders are different, mark them as a cross gender link.
6) Repeat 2) to 5) until all the links are analyzed.
Below is the script implement:

Listing 5: Python script to get gender of each account

```
import os
import json
# import sys
# reload(sys)
# sys.setdefaultencoding('utf-8')

nameFile=open('followers_name.txt')
genderFile=open('gender.txt','w')
droppedlog=open('p2_dropped.txt','w')
strline=nameFile.readlines()
for line in strline:
     try:
          firstname=line.split(' ')[0]
          genderobj=json.load(os.popen('GET https://api.genderize.io/?name="%s"'%
              firstname))
          gender=genderobj['gender']
          if 'null' in gender:
               droppedlog.write(line)
          else:
               genderFile.write(line.strip()+'\t'+gender+'\n')
     except Exception, e:
          droppedlog.write(line)
          continue

nameFile.close()
genderFile.close()
droppedlog.close()
```

Some names are too confusing to guess, so these account are dropped:

```
           ł   ł
------------
Jcanela
Marc Vila Balma a
I2CVB
         i
DPF Manager
TI/A Initiative
WOSP 2015
JCDL Conference
Juneki
Albert Gubern M rida
Ludus Mundi
WOSP 2014
monsorcas
Pontegafas
 Rub n Amaro
JCDL2013
Carlos Garc a Rubio
Nico Hormaz bal
 N ria Vel zquez
ASKS
UdiGitalEdu
Pyramid
```

The gender list we have:

Jose Antonio Olvera	male
Blue Room Innovation	male
Scott Ainsworth	male
candi cadiz munoz	female
Shawn M. Jones	male
Yasmina Anwar	female
Kings web it	male
Carles Serradell	male
Madalina - Carla	female
Ciro Llueca	male
Daniel Ruiz Mestre	male
Rosmery	female
Mohamed Aturban	male
Centre Easy	male
Gina Xifra	female
Mat Kelly	male
Gerard Pons	male
Michael A. Herzog	male
Justin F Brunelle	male
Paulo Carrillo	male
Albert Trias	male
Albert Saavedra	male
ANTONIO OLVERA GAGA	male
La Liga de Futbol	female
It is forever	male
David Plaza	male

Finally, calculate the male and female numbers to perform the homophily test.

Listing 6: python code to test the gender homophily

```
import datetime
gfile=open('gender.txt')
strline=gfile.readlines()
gender={}
nmale=0
nfemale=0
for line in strline:
     tuples=line.split('\t')
     if tuples[1].strip() == 'male':
          nmale=nmale+1
     else:
          nfemale=nfemale+1
     gender[tuples[0]]=tuples[1].strip()
gfile.close()

idfile=open('followers.txt')
namefile=open('followers_name.txt')
namedict={}
idlist=list(idfile.readlines())
namelist=list(namefile.readlines())
```

```
for i in range(len(idlist)):
     namedict[idlist[i].strip()]=namelist[i].strip()
idfile.close()
namefile.close()

resultFile=open('p2_gender homophily.txt','w')
resultFile.write('Generated in %s\n'%datetime.datetime.now())
resultFile.write('Author: Neo <Zetan Li>\n\n')
resultFile.write('Male : %d\n'%nmale)
resultFile.write('Female : %d\n'%nfemale)
p=float(nmale)/float(nmale+nfemale)
q=float(nfemale)/float(nmale+nfemale)
pq=2*p*q
resultFile.write('p = %.2f\nq = %.2f\n2pq = %.2f\n\n'%(p,q,pq))

resultFile.write('Cross gender links:\n')
resultFile.write('-----------------------------------------------\n')
linkfile=open('net.txt')
strline=linkfile.readlines()
ncross=0
nlinks=0
for line in strline:
     tuples=line.split('\t')
     if namedict[tuples[2]] not in gender or namedict[tuples[3]] not in gender :
          continue
     gender1=gender[namedict[tuples[2]]]
     gender2=gender[namedict[tuples[3]]]
     if gender1 != gender2:
          ncross=ncross+1
          resultFile.write('%s <----> %s\n'%(namedict[tuples[2]],namedict[tuples
              [3]]))
     nlinks=nlinks+1
linkfile.close()
resultFile.write('\nSummary of cross gender links: %d out of %d\n'%(ncross,nlinks)
    )
if nlinks>0:
     resultFile.write('Percentage of cross gender links : %.2f'%(float(ncross)/
         float(nlinks)))
resultFile.close()
```

Below is the script output:

```
Generated in 2016-03-17 21:54:21.067811
Author: Neo <Zetan Li>

Male : 22
Female : 7
p = 0.76
q = 0.24
2pq = 0.37

Cross gender links:
-----------------------------------------------
```

```
Jose Antonio Olvera <----> candi cadiz munoz
Jose Antonio Olvera <----> Yasmina Anwar
Jose Antonio Olvera <----> Madalina - Carla
Jose Antonio Olvera <----> Rosmery
Jose Antonio Olvera <----> Gina Xifra
Jose Antonio Olvera <----> Elisabet Blanch
Jose Antonio Olvera <----> La Liga de Futbol
Blue Room Innovation <----> Gina Xifra
Scott Ainsworth <----> Yasmina Anwar
candi cadiz munoz <----> Centre Easy
Shawn M. Jones <----> Yasmina Anwar
Yasmina Anwar <----> Kings web it
Yasmina Anwar <----> Mohamed Aturban
Yasmina Anwar <----> Mat Kelly
Yasmina Anwar <----> Michael A. Herzog
Yasmina Anwar <----> Justin F Brunelle
Kings web it <----> Gina Xifra
Carles Serradell <----> Gina Xifra
Daniel Ruiz Mestre <----> Rosmery
Centre Easy <----> Gina Xifra
Centre Easy <----> Elisabet Blanch
Gina Xifra <----> Paulo Carrillo
Paulo Carrillo <----> Elisabet Blanch
Albert Trias <----> Elisabet Blanch
Albert Saavedra <----> Elisabet Blanch
Elisabet Blanch <----> It is forever
Elisabet Blanch <----> David Plaza

Summary of cross gender links: 27 out of 86
Percentage of cross gender links : 0.31
```

From above, we can see the percentage of cross gender edge is 0.31 ¡ 2pq, ∴ Evidence of homophily

References

[1] Mike Bostock. *Curved Links*, 2016 (accessed March 15, 2016).

[2] genderize.io. *Determine the gender of a first name*, 2016 (accessed March 17, 2016).