PageRank Exploration of the WEBSPAM UK 2007 Collection in Python

Shukry Zablah

March 31, 2020

Contents


```
(setq org-src-tab-acts-natively t)
(setq org-confirm-babel-evaluate nil)
```

```
  (setq org-src-preserve-indentation t)
  (setq org-edit-src-content 0)
```

```
0
```

1 Read the Hostnames

In this secition we read the hostnames file into a hash table called `id2name`.

```
import gzip
import csv

INPUT_HOSTNAMES = "../data/uk-2007-05.hostnames.txt.gz"
id2name = {}
with gzip.open(INPUT_HOSTNAMES, "rt", encoding="utf-8") as input_file:
    reader = csv.reader(input_file, delimiter=' ', quotechar='"')
    for record in reader:
        id2name[int(record[0])] = record[1]
```

And to verify our work, we do some assertions we know to be true already.

```
assert len(id2name) == 114529
assert id2name[801] == 'bacteria.stats.ox.ac.uk'
assert id2name[4778] == 'medphoto.wellcome.ac.uk'
```

Looks like the assertions passed, so we proceed to the next section.

2 Implement PageRank

Below we provide an implementation of pagerank. We have traded speed for memory, as we are interested in processing larger graphs and are limited by RAM usage.

```
def pagerank(path, niter, alpha):
    N = get_N(path)
    score = [1/N for i in range(N)]
    aux = [0 for i in range(N)]
    for i in range(niter):
        with gzip.open(path, "rt", encoding="utf-8") as input_file:
            input_file.readline()
            source = 0
            while source < N:
                line = input_file.readline()
                if line.strip():
                    links = line.split(" ")
                    degree = len(links)
                    for link in links:
                        destination = int(link.split(":")[0])
                        aux[destination] += score[source]/degree
```

```
                source += 1
            for v in range(N):
                score[v] = alpha * aux[v] + (1 - alpha) * (1/N)
                aux[v] = 0
    return score

def get_N(compressedGraphFile):
    with gzip.open(compressedGraphFile, "rt", encoding="utf-8") as input_file:
        return int(input_file.readline())
```

3 Find the nodes with higher PageRank

Now we use our implementation of pagerank to find the scores for the file `uk-2007-05.hostgraph_weighted.graph-`

```
scores = pagerank(path = "../data/uk-2007-05.hostgraph_weighted.graph-txt.gz",
                  niter=20,
                  alpha = 0.85)
```

Now we obtain a list of hosts sorted by their pagerank score. In order to do this we keep track of the scores' indeces in a hash table, and sort the list of scores.

```
sorted_scores = sorted(enumerate(scores), key=lambda x: x[1], reverse=True)

num = 5
for i in range(num):
    print(id2name[sorted_scores[i][0]])
```

```
www.opsi.gov.uk
www.adobe.co.uk
www.ico.gov.uk
www.dti.gov.uk
www.defra.gov.uk
```

The top three commercial hosts are Adobe, BBC, and the National Rail Service.

4 Read a list of spam hosts

In this section we are going to read a list of spam hosts.

```
spam_files = ["../data/WEBSPAM-UK2007-SET1-labels.txt",
              "../data/WEBSPAM-UK2007-SET2-labels.txt"]
is_spam = {}
for spam_file in spam_files:
    with open(spam_file, "rt", encoding="utf-8") as input_file:
        for line in input_file:
            if line.strip():
                tokens = line.split(" ")
                if tokens[1] == "spam":
```

```
                is_spam[int(tokens[0])] = True
            else:
                is_spam[int(tokens[0])] = False
```

5 Run non-spam PageRank

```
def nonspam_pagerank(path, niter, alpha):
    N = get_N(path)
    score = [1/N for i in range(N)]
    aux = [0 for i in range(N)]
    for i in range(niter):
        with gzip.open(path, "rt", encoding="utf-8") as input_file:
            input_file.readline()
            source = 0
            while source < N:
                line = input_file.readline()
                if line.strip():
                    links = line.split(" ")
                    degree = len(links)
                    for link in links:
                        destination = int(link.split(":")[0])
                        try:
                            source_is_spam = is_spam[source]
                        except KeyError:
                            source_is_spam = False
                        try:
                            destination_is_spam = is_spam[destination]
                        except KeyError:
                            destination_is_spam = False
                        will_use = (not (source_is_spam or
                                         destination_is_spam))
                        if will_use:
                            aux[destination] += score[source]/degree
                source += 1
            for v in range(N):
                score[v] = alpha * aux[v] + (1 - alpha) * (1/N)
                aux[v] = 0
    return score

scores_no_spam = nonspam_pagerank(path = "../data/uk-2007-05.hostgraph_weighted.graph-txt.gz",
                                  niter=20,
                                  alpha = 0.85)
```

6 Compute spam gain

In the next section we compute the spam gain, which is defined as the `normal/spam` ratio of the pagerank scores.

```
spam_gain = [normal / spam for normal, spam in zip(scores, scores_no_spam)]

sorted_spam_gain = sorted(enumerate(spam_gain), key=lambda x: x[1], reverse=True)

num = 5
for i in range(num):
    print(id2name[sorted_spam_gain[i][0]])
```

```
www.escortnet.co.uk
www.missionfish.org.uk
www.statistics.006.free-counter.co.uk
www.uk-shoponline.co.uk
www.shop.co.uk
```

7 Compute one variant of PageRank

```
import math
def nonspam_pagerank_with_sqrt_degree(path, niter, alpha):
    N = get_N(path)
    score = [1/N for i in range(N)]
    aux = [0 for i in range(N)]
    for i in range(niter):
        with gzip.open(path, "rt", encoding="utf-8") as input_file:
            input_file.readline()
            source = 0
            while source < N:
                line = input_file.readline()
                if line.strip():
                    links = line.split(" ")
                    degree = len(links)
                    for link in links:
                        destination = int(link.split(":")[0])
                        try:
                            source_is_spam = is_spam[source]
                        except KeyError:
                            source_is_spam = False
                        try:
                            destination_is_spam = is_spam[destination]
                        except KeyError:
                            destination_is_spam = False
                        will_use = (not (source_is_spam or
                                         destination_is_spam))
                        if will_use:
                            aux[destination] += score[source]/math.sqrt(degree)
                source += 1
            for v in range(N):
                score[v] = alpha * aux[v] + (1 - alpha) * (1/N)
```

```
                aux[v] = 0
    return score

scores_wsqrtdeg = nonspam_pagerank_with_sqrt_degree(path = "../data/uk-2007-05.hostgraph_weighte
                                                    niter=20,
                                                    alpha = 0.85)

sorted_scores_wsqrtdeg = sorted(enumerate(scores_wsqrtdeg), key=lambda x: x[1], reverse=True)

num = 5
for i in range(num):
    print(id2name[sorted_scores_wsqrtdeg[i][0]])
```

```
www.dataprotection.gov.uk
www.libdems.org.uk
www.prai.co.uk
islington-libdems.org.uk
warwick-leamington-libdems.org.uk
```

8 Deliverables

8.1 The list of top-20 sites by PageRank

```
num = 20
for i in range(num):
    print(id2name[sorted_scores[i][0]])
```

```
www.opsi.gov.uk
www.adobe.co.uk
www.ico.gov.uk
www.dti.gov.uk
www.defra.gov.uk
news.bbc.co.uk
www.direct.gov.uk
www.dfes.gov.uk
www.fsa.gov.uk
www.nationalrail.co.uk
www.communities.gov.uk
www.bbc.co.uk
www.google.co.uk
www.dh.gov.uk
www.hmso.gov.uk
www.hse.gov.uk
www.fco.gov.uk
www.nationaltrust.org.uk
www.homeoffice.gov.uk
mysite.wanadoo-members.co.uk
```

8.2 The list of top-20 sites by No-spam-PageRank

```
sorted_scores_no_spam = sorted(enumerate(scores_no_spam),
                               key=lambda x: x[1], reverse=True)

num=20
for i in range(num):
    print(id2name[sorted_scores_no_spam[i][0]])
```

```
www.opsi.gov.uk
www.adobe.co.uk
www.ico.gov.uk
www.dti.gov.uk
www.defra.gov.uk
news.bbc.co.uk
www.direct.gov.uk
www.dfes.gov.uk
www.fsa.gov.uk
www.nationalrail.co.uk
www.communities.gov.uk
www.bbc.co.uk
www.google.co.uk
www.dh.gov.uk
www.hmso.gov.uk
www.hse.gov.uk
www.fco.gov.uk
www.nationaltrust.org.uk
www.homeoffice.gov.uk
mysite.wanadoo-members.co.uk
```

8.3 Your comment (1-2 paragraphs) on the similarities/differences between the two lists above

The list of the top 20 sites sorted by PageRank score with and without counting spam show no difference. That means, the popularity of these sites is such, that spam doesn't have a noticeable effect on the ranking of these sites.

8.4 The list of top-20 sites containing .co.uk by PageRank

```
num = 20
count = 0
i = 0
while count < num:
    name = id2name[sorted_scores[i][0]]
    if name.endswith(".co.uk"):
        print(name)
        count += 1
    i += 1
```

```
www.adobe.co.uk
news.bbc.co.uk
www.nationalrail.co.uk
www.bbc.co.uk
www.google.co.uk
mysite.wanadoo-members.co.uk
www.actinic.co.uk
www.networkrail.co.uk
www.caa.co.uk
www.erolonline.co.uk
www.punterlink.co.uk
www.streetmap.co.uk
www.tso.co.uk
www.kelkoo.co.uk
www.guardian.co.uk
www.rac.co.uk
www.event-management-uk.co.uk
www.telegraph.co.uk
www.investorsinpeople.co.uk
www.business-directory-uk.co.uk
```

8.5 The list of top-20 sites containing .co.uk by No-spam-PageRank

```
num = 20
count = 0
i = 0
while count < num:
    name = id2name[sorted_scores_no_spam[i][0]]
    if name.endswith(".co.uk"):
        print(name)
        count += 1
    i += 1
```

```
www.adobe.co.uk
news.bbc.co.uk
www.nationalrail.co.uk
www.bbc.co.uk
www.google.co.uk
mysite.wanadoo-members.co.uk
www.actinic.co.uk
www.networkrail.co.uk
www.caa.co.uk
www.erolonline.co.uk
www.streetmap.co.uk
www.tso.co.uk
www.punterlink.co.uk
www.kelkoo.co.uk
www.guardian.co.uk
```

```
www.rac.co.uk
www.event-management-uk.co.uk
www.telegraph.co.uk
www.investorsinpeople.co.uk
www.business-directory-uk.co.uk
```

8.6 Your comment (1-2 paragraphs) on the similarities/differences between the two lists above

The only difference is that www.punterlink.co.uk fell two spots in the rankings. Again, there is almost no difference between the two ranked lists, which means that these sites probably have a small spam gain.

8.7 The list of top-20 sites containing .gov.uk by PageRank

```
num = 20
count = 0
i = 0
while count < num:
    name = id2name[sorted_scores[i][0]]
    if name.endswith(".gov.uk"):
        print(name)
        count += 1
    i += 1
```

```
www.opsi.gov.uk
www.ico.gov.uk
www.dti.gov.uk
www.defra.gov.uk
www.direct.gov.uk
www.dfes.gov.uk
www.fsa.gov.uk
www.communities.gov.uk
www.dh.gov.uk
www.hmso.gov.uk
www.hse.gov.uk
www.fco.gov.uk
www.homeoffice.gov.uk
www.dft.gov.uk
www.dataprotection.gov.uk
www.dwp.gov.uk
www.legislation.hmso.gov.uk
www.informationcommissioner.gov.uk
www.statistics.gov.uk
www.hm-treasury.gov.uk
```

8.8 The list of top-20 sites containing .gov.uk by No-spam-PageRank

```
num = 20
count = 0
i = 0
while count < num:
    name = id2name[sorted_scores_no_spam[i][0]]
    if name.endswith(".gov.uk"):
        print(name)
        count += 1
    i += 1
```

```
www.opsi.gov.uk
www.ico.gov.uk
www.dti.gov.uk
www.defra.gov.uk
www.direct.gov.uk
www.dfes.gov.uk
www.fsa.gov.uk
www.communities.gov.uk
www.dh.gov.uk
www.hmso.gov.uk
www.hse.gov.uk
www.fco.gov.uk
www.homeoffice.gov.uk
www.dft.gov.uk
www.dataprotection.gov.uk
www.dwp.gov.uk
www.legislation.hmso.gov.uk
www.informationcommissioner.gov.uk
www.statistics.gov.uk
www.hm-treasury.gov.uk
```

8.9 Your comment (1-2 paragraphs) on the similarities/differences between the two lists above

Again, there is no difference between the top 20 government sites when counting or not counting spam. It probably doesn't make sense to think that government sites would have a high spam gain.

8.10 The list of top-20 sites by spam gain

```
num = 20
for i in range(num):
    print(id2name[sorted_spam_gain[i][0]])
```

```
www.escortnet.co.uk
www.missionfish.org.uk
www.statistics.006.free-counter.co.uk
www.uk-shoponline.co.uk
```

```
www.shop.co.uk
www.geordie-girls.co.uk
www.into.demon.co.uk
www.computerarts.co.uk
www.aili.co.uk
connect4fun.co.uk
www.kompass.co.uk
www.mercurywd.co.uk
www.theshopping-centre.co.uk
www.markwarner.co.uk
www.suppliersnearby.co.uk
www.quality-site-finder.co.uk
www.hertfordshiremobilediscos.co.uk
www.eastwoodtoday.co.uk
www.jlc.me.uk
www.ideas21.co.uk
```

8.11 Your comment (1-2 paragraphs) on the list of sites by spam gain

It is clear from the list of sites with the highest values of spam gain that there is a mixture of shady sites such as www.escortnet.co.uk and what seem to be more legitimate sites such as www.computerarts.co.uk. These sites that look more legitimate probably rely a lot in spammy marketing tactics and relationships with spam sites through advertisement.

8.12 A brief description of your variant of PageRank

My variant of PageRank considers the sqrt of the degree instead of the degree. This means that we don't assign the same important to a large degree as the vanilla version of PageRank. Since it is not a log transformation we don't have to worry about having a transformed degree of 0.

8.13 The list of top-20 sites by your variant of PageRank

```
num = 20
for i in range(num):
    print(id2name[sorted_scores_wsqrtdeg[i][0]])
```

```
www.dataprotection.gov.uk
www.libdems.org.uk
www.prai.co.uk
islington-libdems.org.uk
warwick-leamington-libdems.org.uk
libdems4london.org.uk
montlibdems.org.uk
chichesterlibdems.org.uk
surreyheathlibdems.org.uk
bobrussell.org.uk
stevegoddard.org.uk
emilygasson.org.uk
jameskeeley.org.uk
```

bracknell-libdems.org.uk
darren4streatham.org.uk
friendsofstoneymiddletonschool.org.uk
garylawson.org.uk
jamesquinlanforparliament.org.uk
liberty-network.org.uk
lizleffman.org.uk

8.14 Your comment (1-2 paragraphs) on the list of top sites by your variant of PageRank

It is not immediately apparent the relationship of the top rankings of my variant with the vanilla PageRank rankings. Since we don't assign as much of an importance to the degree, that means that we don't expect to see those sites with huge degrees, like google or adobe. However, it is not clear what we can say about these sites without further exploration.