Assignment 4

Fall 2016
CS834 Introduction to Information Retrieval
Dr. Michael Nelson

Mathew Chaney

December 3, 2016

Contents

List of Figures

List of Tables

1 Question 8.3

1.1 Question

For one query in the CACM collection (provided at the book website), generate a ranking using Galago, and then calculate average precision, NDCG at 5 and 10, precision at 10, and the reciprocal rank by hand.

1.2 Approach

Galago version 3.10 was first downloaded from the Project Lemur Source Forge website, which can be found at the following url: https://sourceforge.net/projects/lemur/files/lemur/galago-3.10/.

An index of the CACM corpus, downloaded from the book website http://www.search-engines-book.com/collections/, was created with Galago with the following command:

```
galago build -indexPath=cacm.index -inputPath=docs -server=true
```

This index was used with Galago running as a search engine with the following command:

```
galago search -index=cacm.index
```

The getrel.py script was created to issue queries to the running Galago search server using the Python Requests library [1]. The HTML responses were then parsed using the Python Beautiful Soup library [**?**], where the CACM document identifiers were extracted for use in calculating the different evaluation scores for the Galago ranking.

Precision and Recall were calculated with the following equations:

$$Recall = \frac{\mid A \cap B \mid}{\mid A \mid} \tag{1}$$

$$Precision = \frac{\mid A \cap B \mid}{\mid B \mid} \tag{2}$$

In these equations, A is the relevant set of documents for the query, and B is the set of retrieved documents.

1.2.1 Caclulating Average Precision for CACM Query 1

Query 1 and 10 were used for this question. Average precision was calculated

2 Appendix

2.1 Code listings

```
#!/usr/bin/python

import re
import requests
from bs4 import BeautifulSoup
from pprint import pprint as pp


def buildrel():
    rel = {}
    for line in open('cacm.rel').readlines():
        q, _, doc, _ = line.split()
        if q not in rel:
            rel[q] = []
        rel[q].append(doc)
    return rel


RE = re.compile('/home/mchaney/workspace/edu/cs834-f16/assignments/assignment4/code/cacm/docs/(CACM-[\d]+).html')
ID = {'id':'result'}
URL = 'http://0.0.0.0:{0}/search'
QUERY1 = 'what articles exist which deal with tss time sharing system an operating system for ibm computers'
QUERY10 = 'parallel languages languages for parallel computation'
PDICT = {'q': QUERY1}

def query(query, port=54312):
    PDICT['q'] = query
    res = requests.get(URL.format(port), params=PDICT)
    if not res.ok:
        return None
    soup = BeautifulSoup(res.text, 'html.parser')
    return [RE.match(href.text).groups()[0] for href in soup.select("#result a")]


def recall(rel, retr):
    A = set(rel)
    B = set(retr)
    return float(len(A.intersection(B))) / len(A)


def precision(rel, retr):
    A = set(rel)
    B = set(retr)
    return float(len(A.intersection(B))) / len(B)


def run(rel, retr, func):
    rr = []
    for i in range(1, len(retr)):
        rr.append(func(rel, retr[:i]))
    return rr


def avg(rel, retr, func):
    rr = run(rel, retr, func)
    return sum(rr)/len(rr)


rel = buildrel()

print 'query:', QUERY1
retr = query(QUERY1)
print 'retrieved:', retr
print 'relevant:', rel['1']
print 'precision:', precision(retr, rel['1'])
print 'recall:', recall(retr, rel['1'])
print 'average precision:', avg(retr, rel['1'], precision)
print
print 'query:', QUERY10
```

```
retr = query(QUERY10)
print 'retrieved:', retr
print 'relevant:', rel['10']
print 'precision:', precision(retr, rel['10'])
print 'recall:', recall(retr, rel['10'])
print 'average precision:', avg(retr, rel['10'], precision)
```

Listing 1: getrel.py

```
(skipped debug output)
Stage parsePostings completed with 0 errors.
Stage writeFields completed with 0 errors.
Stage writeNames completed with 0 errors.
Stage writeLengths completed with 0 errors.
Stage writeNamesRev completed with 0 errors.
Stage writeCorpusKeys completed with 0 errors.
Stage writePostings completed with 0 errors.
Stage writePostings-krovetz completed with 0 errors.
2016-12-02 07:30:20.267:INFO:oejs.ServerConnector:main: Stopped ServerConnector@3b4d8f47{
    HTTP/1.1,[http/1.1]}{0.0.0.0:50935}
Done Indexing.
  - 0.00 Hours
  - 0.18 Minutes
  - 11.01 Seconds
Documents Indexed: 3204.
```

Listing 2: Output from building the CACM index.

```
[mchaney@mchaney-l getrel]$ python getrel.py
query: what articles exist which deal with tss time sharing system an operating system for
    ibm computers
retrieved: [u'CACM-1410', u'CACM-1827', u'CACM-0397', u'CACM-1280', u'CACM-2319', u'CACM
    -1938', u'CACM-1908', u'CACM-1885', u'CACM-1071', u'CACM-2535']
relevant: ['CACM-1410', 'CACM-1572', 'CACM-1605', 'CACM-2020', 'CACM-2358']
precision: 0.2
recall: 0.1
average precision: 0.520833333333

query: parallel languages languages for parallel computation
retrieved: [u'CACM-2785', u'CACM-1811', u'CACM-1262', u'CACM-0950', u'CACM-2895', u'CACM
    -2700', u'CACM-2851', u'CACM-1747', u'CACM-2289', u'CACM-2266']
relevant: ['CACM-46', 'CACM-141', 'CACM-392', 'CACM-950', 'CACM-1158', 'CACM-1198', 'CACM
    -1262', 'CACM-1380', 'CACM-1471', 'CACM-1601', 'CACM-1613', 'CACM-1747', 'CACM-1795', '
    CACM-1811', 'CACM-2060', 'CACM-2150', 'CACM-2256', 'CACM-2289', 'CACM-2342', 'CACM
    -2376', 'CACM-2433', 'CACM-2618', 'CACM-2664', 'CACM-2685', 'CACM-2700', 'CACM-2714', '
    CACM-2777', 'CACM-2785', 'CACM-2851', 'CACM-2895', 'CACM-2896', 'CACM-2912', 'CACM
    -3039', 'CACM-3075', 'CACM-3156']
precision: 0.228571428571
recall: 0.8
average precision: 0.155878721569
```

Listing 3: Output from running the getrel.py script for queries 1 and 10 from the CACM collection.

3 References

[1] Kenneth Reitz. Requests: HTTP for Humans. Available at http://docs.python-requests.org/en/master/. Accessed: 2016/09/20.