

資訊檢索與文字探勘導論

hw4 Report

B03705012 張晉華

1. 程式語言

Python 3.5.2

2. 執行環境

Linux OS (Ubuntu 16.04 LTS)

Python3 Packages Requirements:

- Python Natural Language Toolkit(nltk 3.2.2)
- Python Numpy(1.13.3)

3. 執行方式

● Package Installation

◆ Python Natural Language Toolkit(NLTK)

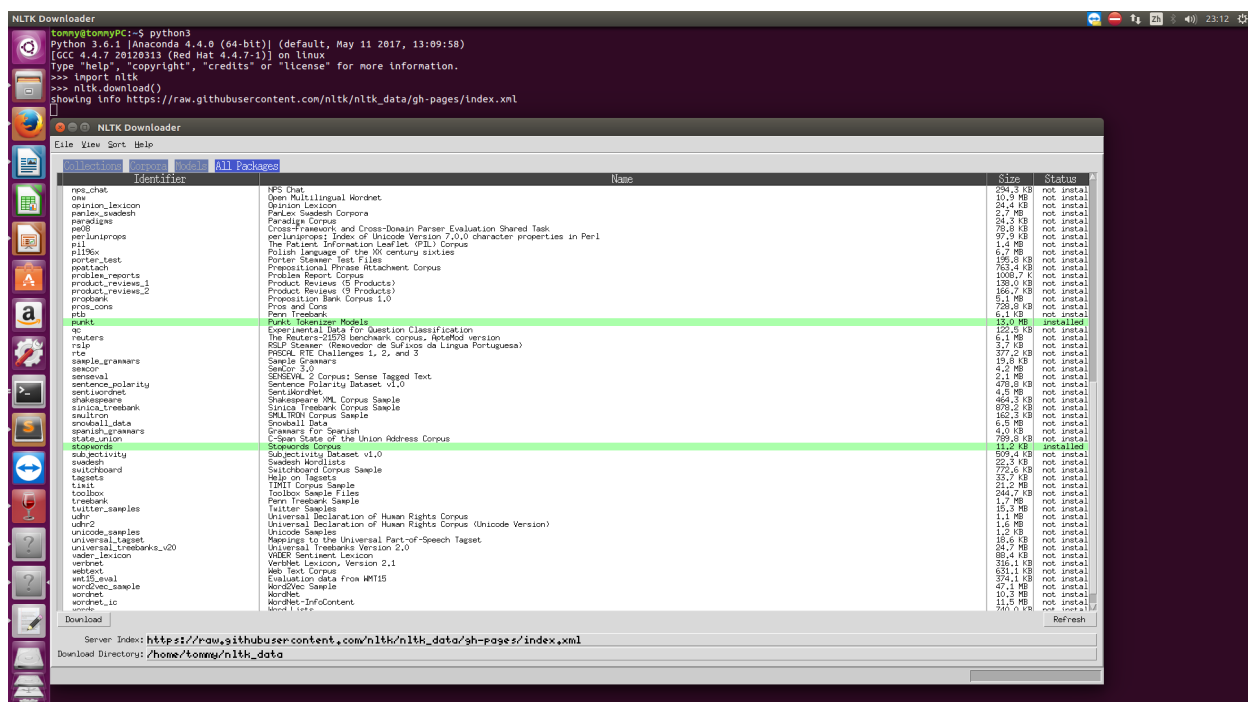
Step1 : Install Packages

(可以透過 pip3 install nltk 安裝)

Step2 : Corpus Download

有兩個 Corpus 需要下載以供程式使用：punkt(tokenize 需要)和 stopwords

可以在 python3 環境下執行 nltk.download() 下載



- ◆ Python Numpy：可透過 `pip3 install numpy` 安裝
- 執行指令
 - ◆ 首先確定同路徑內有 `data/IRTM/(1095 篇文檔)`、`my_tokenizer.py`、`my_dictionary.py`、`my_cosine.py`、`cluster.py` 4 個項目
 - ◆ `cd` 進程式所在路徑，在 `terminal` 執行 `python3 cluster.py`，檔案結果產出在同一路徑的 `8.txt`、`13.txt`、`20.txt`，內容為 3 個 K 值的 `cluster` 結果。

4. 作業處理邏輯說明

- `cluster.py`

分為 `pre_process`, `cluster`, `output` 3 個大步驟

- `pre_process`

利用 `hw2` 的 `code` 建立好每個文章的 `normalized tfidf vectors`. 並且算好每兩篇文章之間的 `cosine similarity` 並丟進 `heap` 內排序。
- `cluster`
 - 合併部份
 - 利用並查集的結構優化，用一張表紀錄這個文章被併入哪個文章的群中（如果沒有則值為自己），以及自己本身又併入了多少的文章，如果需要尋找這個文章群的根則利用遞迴就可以找到。
 - `Single Link`
 - 將 `pre_process` 處理好的 `heap` 依序取出，若取出的兩個文章目前屬於不同群，則代表他們會是現存所有群之中 `single link` 最短的，則將兩群合併，重複此步驟直到現存文章群的數量=K。
 - `Centroid`
 - 一樣從 `heap` 依序取出，若取出的兩個文章目前屬於不同群則合併，而合併後要將根節點的文章 `vector` 更新成合併後新的 `centroid`（用根結點代表新群的 `centroid`），並且刪去原本 `heap` 內所有有取出的這兩個文章的點對，並加入新算出來的 `centroid` 與其他文章群根節點的 `cosine similarity`，重複此步驟直到現存文章群的數量=K。
- `output`

利用遞迴的方式找到每一篇文章在並查集內的根節點，同一個根節點的文章就代表在同一個群裡面，再將結果輸出成 `txt` 檔。

執行截圖如下頁圖：

```
tommy@tommyPC: ~/NTU/IR2017/hw4
tommy@tommyPC:~/NTU/IR2017/hw4$ python3 cluster.py
Documents Total : 1095
Number of Terms : 13765
Save dictionary.txt
Save Vectors of Document: 1095
Time consumed : 22s
Iteration 1 : merge 704,705
Iteration 2 : merge 100,105
Iteration 3 : merge 661,662
Iteration 4 : merge 47,48
Iteration 5 : merge 194,228
Iteration 6 : merge 194,229
Iteration 7 : merge 563,564
Iteration 8 : merge 563,594
Iteration 9 : merge 563,595
Iteration 10 : merge 731,732
Iteration 11 : merge 791,795
Iteration 12 : merge 620,621
Iteration 13 : merge 475,476
Iteration 14 : merge 925,927
Iteration 15 : merge 942,943
Iteration 16 : merge 304,308
Iteration 17 : merge 7,8
Iteration 18 : merge 210,211
Iteration 19 : merge 210,212
Iteration 20 : merge 847,848
Iteration 21 : merge 526,528
Iteration 22 : merge 190,191
Iteration 23 : merge 1033,1035
Iteration 24 : merge 499,500
Iteration 25 : merge 154,157
Iteration 26 : merge 831,837
Iteration 27 : merge 506,507
Iteration 28 : merge 582,583
Iteration 29 : merge 242,243
Iteration 30 : merge 1084,1087
Iteration 31 : merge 675,676
Iteration 32 : merge 680,683
Iteration 33 : merge 329,332
Iteration 34 : merge 197,199
Iteration 35 : merge 895,896
Iteration 36 : merge 854,855
Iteration 37 : merge 821,823
Iteration 38 : merge 99,101
Iteration 39 : merge 196,201
Iteration 40 : merge 519,522
Iteration 41 : merge 604,606
Iteration 42 : merge 824,827
Iteration 43 : merge 735,736
Iteration 44 : merge 498,499
Iteration 45 : merge 490,492
Iteration 46 : merge 898,905
Iteration 47 : merge 371,380
Iteration 48 : merge 840,841
Iteration 49 : merge 968,969
Iteration 50 : merge 53,54
Iteration 51 : merge 87,88
Iteration 52 : merge 820,821
Iteration 53 : merge 910,911
Iteration 54 : merge 953,954
```

由於 Iteration 較多無法完整截圖，執行上程式會印出每次 Iteration 合併的兩個文章群根節點的編號，由於 K 值最小是 8，因此程式會在 Iteration 1087 停止。

5. 心得

這次的作業實作了兩種 Similarity measure between clusters 的方法：Single Link 和 Centroid，而兩個作法的結果截然不同，Single Link 容易出現大者恆大的現象，結果出來會容易形成一兩個大群配上其他只有一兩篇文章的群，而 Centroid 相對來說就較為集中，因此這次最後我選擇用 Centroid 的方法，如果以後有需要再實作 clustering 的算法時，應該要多注意多思考，哪種方法表現會比較好，避免用錯方法導致表現不佳的情況出現。