B10615023 楊傑安

NTUST_Information_Retrieval_and_Application_HW1

使用工具

- 語言: Python 3.8.5
- package:
    - tqdm
    - math

設計架構

1. 先將 50 條 query 所含的 127 個字建立成 vocabulary
2. 使用 1. 得到的 vocabulary 算出 query 和 document 的 term frequency
3. 使用 1. 得到的 vocabulary 算出 document 的 document frequency
4. 對其中一條 query, 利用 2. 3. 所得
    a. 計算 向量q
    b. 計算每一個 document 的 向量d
    c. 計算 b. 中的每一個 向量d 和 向量q 的 cosine similarity
    d. 將 documents 依照 c. 所得倒序排列, 即得到此 query 的 revelant documents
5. 重複 4. 即得所有 query 的 revelant documents

困難與心得

算是單純的一個作業, 難點在 document 的數量不小. 如果用所有的字當作 vocabulary 的話, 單靠 cpu 計算會需要很長的一段時間. 我的方法是只使用所有 query 出現的字當作 vocabulary, 因爲只有 127 個 terms, 計算時間大幅減少, 同時還能保證 query 的 term frequency != 0, 不會造成計算上的問題.

另外有一個修改: 計算 idf 時, 因爲出現在 query 的 term 不一定會出現在 document (i.e. df == 0), 在有關 inverse document frequency 的計算時會出現除 0 的情況. 我的解決方法是參考probabilistic model 中的一個做法: 將分母加上 0.1, 避開除 0 的情況