WM Programming HW5 Report

R07922134 陳紘豪

1. 相關使用套件 & 程式執行方法

套件：

```
jieba==0.39
pandas==0.23.4
numpy==1.14.3
```

執行方法：

```
$ python3 preprocess.py -i $1 -c $2

# $1: path to inverted_file.json provided by TAs
# $2: path to NC_1.csv
# 先執行 preprocess.py 來重新計算 tf, idf, document length, inverted file 等 VSM 所需要的資料

$ python3 main.py -q $1 -c $2 -o $3

# $1: path to QS_1.csv
# $2: path to NC_1.csv
# $3: path to the output csv file
# 再來執行 main.py，透過 VSM 和 Pseudo Rocchio Feedback 來產生最後檢索的結果
```

2. 使用的方法

I. Vector Space Model

$$\sum_{t \in Q,D} ln \frac{N - df + 0.5}{df + 0.5} \cdot \frac{(k_1 + 1)tf}{(k_1(1 - b + b\frac{dl}{avdl})) + tf} \cdot \frac{(k_3 + 1)qtf}{k_3 + qtf}$$

使用了以上的公式來計算 IDF, TF, query TF，其中參數的設定為 b = 0.75, k1 = 1.2, k3 = 500，會採取這個公式來實作 VSM 的原因是因為上一次的程式作業使用的效果還不錯，所以就在這次的比賽中繼續使用。

II. Pseudo Rocchio Feedback

$$\vec{q}_m = \alpha \vec{q} + \frac{\beta}{|D_r|} \sum_{\forall \vec{d}_j \in D_r} \vec{d}_j - \frac{\gamma}{|D_n|} \sum_{\forall \vec{d}_j \in D_n} \vec{d}_j$$

PRF 的實作採用以上公式，因為是 Pseudo 的關係，所以我的假設是在第一次檢索的結果中，前 30 名的文件是相關文件，第 301 ~ 330 名的文件為不相關文件，透過這 60 個文件

來對原始的 query 做 query expansion，更新其中的 query TF，參數設定為 alpha = 0.6, beta = 0.3, gamma = 0.1

3. 嘗試過的實驗

最一開始的時候，我是直接採用 language model 來實作，把每個 term 當作是一個 unigram，然後從 inverted_file.json 中去推算出每個 document 背後的 language model，去計算 query likelihood P(Q|D) 並依照分數高低去對 document 排名，最一開始在 public leaderboard 上只有 4 % 而已，發現到是因為我沒有實作 smoothing 的結果，導致整體機率會被一些 stopwords 給 dominate，後來去對 language model 去實作 smoothing 後最高有來到 19.9 %，只是因為我不知道要再如何改善了，就去實作 VSM + PRF 來看看結果如何，結果想不到在 Public 上有 28.77725 %，比 language model 高很多。